Selecting Representative Sample Traces from Large Event Logs

Gaël Bernard
University of Toronto
Faculty of Information
Toronto, Canada
gael.bernard@utoronto.ca

Periklis Andritsos
University of Toronto
Faculty of Information
Toronto, Canada
periklis.andritsos@utoronto.ca

***Abstract*—When event logs are large, the time needed to analyze them using process mining techniques can become prohibitive. In this paper, using sampling, we aim to reduce the size of event logs to p-traces, while minimizing the Earth Movers' Distance (EMD) from the unsampled original event log. We contribute by formalizing log sampling in a canonical form and show its link with the EMD, a metric increasingly used for process mining. Next, we propose three log-sampling algorithms that we evaluate using a collection of 18 event logs from industry. We show that our approach largely reduces the EMD compared to existing sampling strategies. Moreover, we highlight that sampled event logs with low EMDs tend to have better behavioural quality, highlighting the generality of our work.**

***Index Terms*—process mining, sampling, earth mover's distance, stochastic language**

I. Introduction

By turning event logs into information, process mining helps organizations to improve their business processes. Process mining capabilities include discovering process models from event logs, checking conformance, and performing root cause analyses, to name a few. Although these compelling capabilities, extracting insights from large event logs is challenging since some techniques cannot be run using conventional hardware [1]. When they do, the execution time limits the number of techniques and parameters that can be tested. It is a significant limitation given the exploratory nature of most process-mining projects [2].

We claim that these technical issues can be solved by working with a limited but representative subset of traces. Not only does this have the potential to streamline the application of advanced techniques, but it can also reduce the need for a manual inspection of event logs. As Leemans et al. state: *"Analyzing the parts of the log that deviate from the model is tedious: all trace variants are visualised one-by-one, and for larger event logs, it is infeasible to derive information from this view"* [3]. Thus, sampling event logs would allow a business analyst to study a limited number of traces and still grasp the overall behaviour. However, sampling is challenging because it *"quickly introduces under- and oversampled behaviour in event logs, which can be problematic for frequency-based algorithms,"* according to Knols and van der Werf [4].

We use the term p-traces to denote a subset event log of size p. In this paper, we aim to return the p-traces that minimize the Earth Mover's Distance (EMD) from the unsampled event logs. The EMD is a metric that measures the dissimilarity between two multi-dimensional distributions. Incorporating this distance in the sampling objective allows returning a general-purpose summary of the event logs useful for various process mining tasks (e.g., discovery, replay, visualization). In Fig. 1, we illustrate two p-traces and how the EMD helps to pick the best one.

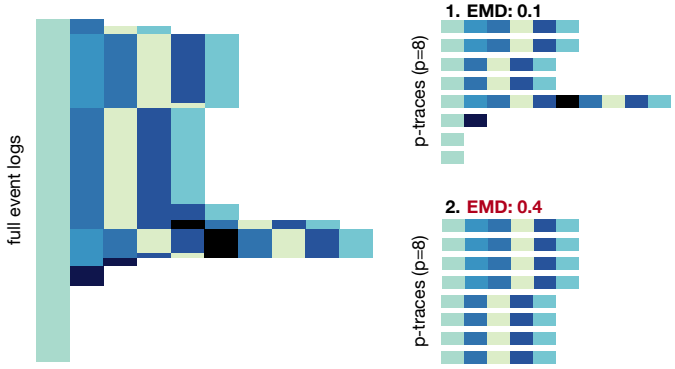

Fig. 1. An illustrative event log downsized to 8-traces in two alternatives ways. The first one is better because it has a lower EMD.

We make the following contributions:

1) We formalize the problem of sampling event logs to p-traces while minimizing the EMD using a linear programming formulation.
2) Since the problem is NP-hard, we propose three sampling heuristics.
3) We benchmark 9 sampling techniques, for 8 values of p, using 18 real datasets, and show that our sampling techniques significantly outperform the existing ones.
4) We highlight the strong correlation between the EMD and the behavioural quality of event logs.

The rest of the paper is organized as follows. We introduce the notation in Section II and the related works in Section III. In Section IV, we formally define the sampling problem, and in Section V we propose three ways to solve it. In Section VI, we benchmark several sampling techniques. We provide a discussion in Section VII and conclude in Section VIII.

II. Preliminaries

We focus on the activity order, so we treat event logs as multisets as introduced in [5]. Table I provides an overview of the notation used throughout the paper.

TABLE I
Notation

Notation	Name	Example
L	event log	$L_1 = [abc^2, abcd^3, ea]$
L^*	variants	$L_1^* = [abc, abcd, ea]$
L^p	p-traces	$L_1^p \subset L_1$ and $\lvert L_1^p \rvert = p$
$stoch(L)$	stochastic lang.	$stoch(L_1) = [abc^{\frac{1}{3}}, abcd^{\frac{1}{2}}, ea^{\frac{1}{6}}]$
$eo(L,p)$	expected occ.	$eo(L_1, 2) = [abc^{\frac{2}{3}}, abcd^{1}, ea^{\frac{1}{3}}]$

Event Log. Let $\mathscr{A}$ be a set of activity names. A trace, σ, is a sequence of activities, i.e., $\sigma \in \mathscr{A}^*$. An event log, L, is a multiset of traces over $\mathscr{A}$; e.g., $L_1 = [\langle a,b,c\rangle^2, \langle a,b,c,d\rangle^3, \langle e,a\rangle]$. For conciseness, we shorten this notation as follows: $[abc^2, abcd^3, ea]$.

Variants and Multiplicity. The variant, L^*, is the set of unique sequences from L (also called the support). The multiplicity of a variant is the number of times it appears in L (its exponent).

Event Log Size. The size of the event log, $|L|$, is the total number of traces it contains, i.e., the sum of the variants' multiplicity.

p-Traces. Let L^p be a subset of L of size p where $L^p \subset L$, $|L^p| = p$, and $0 < p < |L|$. We use the terms p-traces and L^p interchangeably.

Stochastic Language. The stochastic language expresses each variant's probability. Let the stochastic language be a function, $stoch()$, that, given L, returns a real-valued multiset by dividing each variant's multiplicity by $|L|$. For instance, the stochastic language of $[abc^2, abcd^3, ea]$ is $[abc^{\frac{1}{3}}, abcd^{\frac{1}{2}}, ea^{\frac{1}{6}}]$.

Expected Occurrence. Given a stochastic language and the number of desired traces, p, the expected occurrence is the number of times we expect to observe the variants in L^p. More formally, the expected occurrence, $eo()$, is a function that, given L and p, returns a real-valued multiset expressing the expected multiplicity of each variant by multiplying the stochastic language by p. For instance, if the stochastic language is $[abc^{\frac{1}{3}}, abcd^{\frac{1}{2}}, ea^{\frac{1}{6}}]$, the expected occurrence of $L^{p=3}$ is $[abc^1, abcd^{\frac{3}{2}}, ea^{\frac{1}{2}}]$.

III. Related Work

We first position our work and, then we present the existing sampling techniques.

A. Positioning

In [2], the authors highlight the difference between variant-based and trace-based sampling. The primer only returns unique variants, while repetitions of traces are allowed in trace-based sampling. For instance, the two samplings reported in Fig. 1 are trace-based because they both contain some duplicates. Variant-based sampling is particularly relevant for process mining techniques that do not consider the frequency of traces, such as the alpha miner [6] or the inductive miner [7]. By eliminating identical traces, such sampling strategies can drastically reduce event logs' size. However, they are unlikely to produce representative samples. For instance, if a variant happens half of the time but reported only once in L^p, it introduces a critical bias. Because we aim to discover representative sample traces, we focus this work on trace-based sampling.

A closely related topic to log sampling is the removal of outliers traces or activities as presented in [8], [9]. By removing 'noises,' these techniques improve the quality of the discovered process models [8]. However, the 'noises' may be a salient characteristic of the event logs. Therefore, removing all of them to return only the 'happy paths' may not be desirable. For instance, it could lead the process analyst to draw over-optimistic conclusions about the discrepancies between event logs and process models.

Log sampling techniques are also used to approximate conformance analysis [10], [11]. Without replaying the overall event logs, it is possible to approximate the replay with an upper bound [11]. Sampling techniques in this area are especially relevant, and we report them in the next section (Section III-B). It is also worth mentioning that the EMD is used in [3] as a conformance analysis technique.

One sampling strategy is to iteratively grow the number of traces until an objective criterion is met. Bauer et al. propose a statistical framework to perform this task using statistical tests as a stopping criterion [10], [12]. Berti et al. use a similar sampling strategy that stops when the pair-wise activity dependencies resemble the original event logs [13]. These techniques appeal because they come with statistical guarantees that the sampled event logs contain enough information. However, it comes at the price of not being able to control the number of desired traces. Moreover, these techniques do not have built-in functionality to select the most suitable traces; i.e., they do not aim to select the fewest traces possible. Hence, we consider that these contributions complement sampling techniques.

The event logs behavioural characteristic is mentioned in several related works [2], [4], [14]. It is used to designate the directly-follow occurrence of activities. In particular, Knols et al. propose several metrics to measure the behavioural quality of log sampling [4]. These metrics compare the directly-follow occurrence of activities in the original event logs from the sampled ones–using a normalization step to make them comparable. If the difference lies within a predefined bandwidth, the behaviour is considered well sampled. Otherwise, it is under-or over-sampled. The *percentage of truly sampled behaviour* is one of the metrics proposed in [4]. In the experiment, we report its complement that we named the ratio of *Erroneously Sampled Behaviours* (ESB). We favour the complement because it is a distance function like the EMD, and it eases the interpretation of the results; i.e., for both metrics, the lower, the better.

Next, we introduce the existing sampling techniques.

B. Existing Sampling Techniques

The techniques numbered in this section are implemented in the evaluation section (Section VI).

1) Random: Random sampling consists of drawing p samples with replacement from the stochastic language. This approach is described in several works [2], [11], [14], [15]. Using the law of large numbers, it should produce perfect sampling when p is large enough. It is certainly the most natural way to sample an event log and, hence, we consider it as the baseline technique.

2) Stratified Random: Van der Werf et al. suggest treating each variant as a stratum so that stratified sampling can be used to sample event logs [14]. The frequency of a specific variant is also a sorting criterion mentioned in [11], [16]. One of the challenges linked to applying stratified sampling is due to the unbalanced nature of the strata. Indeed, it is common to have few variants with high repetitions and variants appearing only once in the entire event log. For instance, consider the log $[abcd^{90}, abe^{8}, ab^{1}, abcde^{1}]$ and $p = 10$, the following question arose: how many traces to pick in each stratum? Using a multistage approach as described in [14] solves this issue.

3) Variant Biased: As the name suggests, biased sampling favours a particular type of variant. Sani et al. propose to rank the traces and extract the top p ones [2]. They propose various ranking strategies based on the variants' lengths or counts. Since our goal is to propose a fair way to select the traces, biased sampling is unlikely to perform well on metrics measuring logs' representativeness, such as the EMD. Nevertheless, we purposefully added such a technique in the evaluation section for benchmarking purposes.

4) Behaviour-based: In [2], the authors extract the most prominent and rarest behaviours based on the directly-follow occurrences. Each variant receives, respectively, positive or negative points when they contain such behaviours. The sum of points gained by a variant is then normalized by its length. Finally, one can sample the event logs by taking the p variants that accumulated the maximum number of points.

5) LogRank: In [17], Liu et al. propose a technique inspired by the well-known PageRank algorithm [18], which Google used to measure the importance of a webpage by analyzing its incoming hyperlinks from other web pages. In a process mining context, the webpages are the variants, and a measure of similarity between variants replaces the hyperlinks. We can then return the p most relevant variants.

6) Redundancy Check: Although used in social science to describe life's trajectories, the approach described by Gabadinho et al. is also relevant in a process mining context [16]. It consists of ranking the variants by centrality, i.e., the closest to all the other ones. Then, the most central variants are iteratively selected. However, before choosing a trace, one must check that it exceeds a minimum distance threshold from the already selected variants. Doing this ensures diversity in the sampling and is what the authors called the 'redundancy check.'

In the next section, we formalize the representative log sampling task.

IV. Problem Definition

The goal is to build an L^p event log close to the expected occurrence (Section II). The main challenge is that the variant's multiplicity is restricted to integers because a variant cannot partly exist in an event log. To quote: *"only a natural number of traces can be added to a sample, a trace can only be added fully or not at all"* [14]. In opposition, the expected occurrence is a real-valued multiset. For instance, if a variant occurs only once in L it will ineluctably be either under- or over-represented in L^p. In this section, we formulate this problem that we named the "constant capacitated p-median problem". Before presenting it, we introduce two prerequisites: the Levenshtein distance and the Earth Mover's Distance.

A. Normalized Levenshtein Distance

The Levenshtein distance, [19], counts the number of operations (i.e., deletions, insertions, and substitutions) needed to match two sequences. Typically, the distance between abc and $abbd$ is 2 (e.g., 1 insertion of d, 1 substitution of $c \rightarrow b$). In this work, we normalize the Levenshtein distance by the length of the longer sequences to get a distance from 0 to 1, which reflects existing practice in process mining (e.g., [3], [20]).

B. Earth Mover's Distance (EMD)

Informally, the EMD expresses the amount of work needed to move several 'piles of dirt' into various 'holes' while considering the distance between them. More formally, this metric measures the dissimilarity between two multi-dimensional distributions. It was introduced by Rubner et al. for image retrieval in databases [21]. Recently, it was brought in the process mining space by Leemans et al., who uses it to measure the distance between process models and event logs [20]. The EMD uses two important concepts, namely the cost matrix and the reallocation matrix.

1) Cost Matrix (c): The cost matrix reports the normalized Levenshtein distances between all the traces. Since the diagonal expresses the distance between a trace and itself, its value is zero. As an example, the cost matrix of $[abc, ab, abcdee]$ is:

$\mathbf{c_1}$	abc	ab	$abcdee$
abc	0	$1/3$	$1/2$
ab	$1/3$	0	$2/3$
$abcdee$	$1/2$	$2/3$	0

2) Reallocation Matrix (r): The reallocation matrix is a function that expresses one possible way to transform a stochastic language into another. Dirt can only move, not disappear. Hence the sum of each row should equal the sum of the first language, and the sum of columns should equal the second language. As an illustration, let $\hat{L_1} = [abc^{0.5}, ab^{0.4}, abcdee^{0.1}]$ and $\hat{L_2} = [abc^{0.7}, ab^{0.3}]$ be two stochastic languages. Two examples of reallocation matrices are:

$\mathbf{r_1}$	abc	ab	$abcdee$
abc	.5	.0	.0
ab	.1	.3	.0
$abcdee$	.1	.0	.0

$\mathbf{r_2}$	abc	ab	$abcdee$
abc	.5	.0	.0
ab	.2	.2	.0
$abcdee$	.0	.1	.0

Finally, the EMD is the inner product between the cost matrix (c) and the reallocation matrix (r):

$$emd(\hat{L_1}, \hat{L_2}) = c \cdot r \tag{1}$$

Theoretically, the EMD is calculated using the optimum reallocation matrix, which, in practice, is found using the minimum cost flow algorithm as well as additional heuristics (e.g., [22]).

The EMD provides a convenient way to measure the distance between an event log and one of its downsized versions. We continue the formulation of the problem to put the selection of the best traces into the equation.

C. Constant Capacitated p-Median Problem

We propose the *constant p-median capacitated problem* inspired by the p-median problem [23] and the capacitated p-median problem [24]. The goal is to select the subset of traces that minimize the EMD. We call the selected subset of traces the *representative traces* since their purpose is to represent non-selected traces assigned to them.

Let X be a binary vector of size $|L|$ that indicates which traces are representatives; i.e., which ones are returned by the sampling techniques; and let Y be a binary matrix of size $|L|^2$ that maps traces to representatives. Equipped with these two variables, we can introduce the constant capacitated p-median problem.

Minimize:

$$\sum_{i \in L} \sum_{j \in L} c_{ij} \cdot Y_{ij} \tag{2}$$

Subject to:

$$\sum_{j \in L} X_j = p \qquad \text{(Constraint 1)}$$

$$\sum_{j \in L} Y_{ij} = 1 \qquad \forall i \qquad \text{(Constraint 2)}$$

$$\sum_{i \in L} Y_{ij} \leq \left\lceil \frac{|L|}{p} \right\rceil \qquad \forall i \qquad \text{(Constraint 3)}$$

$$Y_{ij} - X_j \leq 0 \qquad \forall i \in L, \forall j \in L \qquad \text{(Constraint 4)}$$

$$Y_{ij}, X_j \in \{0, 1\} \qquad \text{(Constraint 5)}$$

Where:

p	number of representatives to discover
c_{ij}	cost matrix between traces
Y_{ij}	binary assignment of traces to representatives
X_j	binary selection of representatives traces

The function we aim to minimize involves the cost matrix and the assignment of traces to representatives. It is hence equivalent to the EMD. The first constraint fixes the number of representatives to p. The second constraint ensures that all traces are assigned to one representative. The third constraint imposes a capacity limit to representative traces proportional to p. This capacity limit is what makes this approach different from the existing p-median problems. We round to ceil to ensure that we have enough capacity for all traces and that we can hence satisfy the second constraint as well. The fourth constraint restricts the allocation of traces to representatives, i.e., trace i can be represented by trace j *iff* $X_j = 1$. Finally, the fifth constraint defines that Y and X are limited to binary values.

Linear programming offers a convenient way to describe and eventually solve the problem of event log sampling. In practice, its applicability is limited due to time constraints since the p-median problem is NP-hard [25]. In practice, the algorithms halt when the number of variants exceeds a few hundred variants, which is the case for all the event logs considered in this paper.

The problem is close to the capacitated p-median problem as introduced in [24]. However, the difference lies in the fact that the traces' capacity is constant–proportionally to the number of p–so that each representative traces have the same importance. In the next section, we propose several heuristics that leverage this singularity.

V. Proposed Approaches

This section introduces three ways to solve the constant capacitated p-median problem.

A. Iterative c-min

We named our first approach the *iterative c-min* described in Algorithm 1. The 'c' stands for the maximum capacity of the representatives, i.e., the number of traces a representative is responsible for, and is a direct application of the second constraint from the constant capacitated p-median problem. Once c is defined, the algorithm leverages the cost matrix (Section IV-B1) to iteratively select the representative trace with the smallest distance with the c closest neighbourhoods. Then, these closest neighbourhoods are removed from the cost matrix before looking for the next best representative. The process goes on until we get p-traces.

Algorithm 1 Iterative c-min

Input L, n ▷ L: event log, n: desired number of traces
Output repTraces ▷ repTrace is a subset of L

$repTraces \leftarrow \emptyset$
$cost \leftarrow$ EditDistance(L) ▷ Matrix of costs between traces
$c \leftarrow$ Ceil(Length(L)/n) ▷ Representatives' capacity constraint
while Length($repTraces$) $< n$ **do**
  $minSum \leftarrow \infty$
  for all $trace_i \in L$ **do**
    $closestInd \leftarrow$ cMinInd($c, cost[trace_i]$) ▷ c closest indices
    $sumDist \leftarrow$ Sum($cost[trace_i][closestInd]$)
    if $sumDist < minSum$ **then**
      $minSum \leftarrow sumDist$
      $bestRep \leftarrow L[trace_i]$
      $bestClosestInd \leftarrow closestInd$
  $repTraces$.Add($bestRep$)
  for all $i \in bestClosestInd$ **do**
    $cost[i][:]$.Remove(i) ▷ remove rows
    $cost[:][i]$.Remove(i) ▷ remove columns
    L.Remove(i)
return $repTraces$

B. Expected Occurrence Reduction (EOR)

We aim to reduce the computation time by extracting 'easy' representatives. For instance, if a variant happens half of the time, we want to get it approximately (due to rounding error) half of the times in the sampled event log. We propose a heuristic named the Expected Occurrence Reduction (EOR) that automatically extracts these representatives.

The EOR uses the expected occurrence defined in Section II and applies a two-step process. In step 1, it creates an initial sampling set with the integer part of the expected occurrence. Then, step 2 completes the initial set (so that $|L^p| = p$) with the fractional part using any sampling strategies. For instance, if the expected occurrence is $[abd^{4.8}, abc^{0.6}, eaf^{0.6}]$ and $p = 6$, we extract the set $[abd^4]$ in step 1. In step 2, we complete $[abd^4]$ with two additional traces drawn from $[abd^{0.8}, abc^{0.6}, eaf^{0.6}]$.

By definition, in step 2, the expected occurrence will be lower than one, i.e., we do not want to return duplicates in step 2. Ultimately, it means that we can apply variant-based sampling techniques in step 2 for a problem that is initially better suited for trace-based sampling techniques. Interestingly, it corroborates a statement from Sani et al.: *"[variant-based] methods can easily be extended to trace-based sampling methods"*. With EOR, we provide a concrete way to do it.

C. Iterative c-min (Eucl.)

One of the significant bottlenecks of the iterative c-min algorithm is the time and space complexity of the cost matrix, which grows with the number of variants. In this new version of the algorithm, we avoid this step by working in the Euclidean space. To work in this space, we count the n-grams of size one and two; e.g., $[aab]$ becomes $[a^2, b^1, aa^1, ab^1]$. This approach is common in the process mining space (see [17]). Then, we use singular value decomposition (SVD) to reduce the number of dimensions. Finally, one can apply a nearest neighbour search to retrieve the c closest variants. The rest of the algorithm remains unchanged.

VI. Evaluation

We applied 9 sampling techniques, tested 8 values of p ranging from 5 to 200, on 18 datasets, and repeated the experiment 4 times due to the non-deterministic nature of the sampling techniques. Altogether, it results in $9 \cdot 8 \cdot 18 \cdot 4 = 5\,184$ sampling solutions that we evaluated using the EMD (Section IV-B) and the ESB (Section III-A). For the ESB, we used the bandwidth suggested by its authors, i.e., 0.05. Besides the ESB and the EMD, we also recorded the time needed to complete the sampling. We set a maximum sampling duration time limit to 10 minutes because we believe that a longer preprocessing time would hinder the benefit of sampling in the first place. We ran the experiment using a machine with 16GB of RAM, 4 CPUs, and a processor speed of 2.8 GHz.

First, we list the 9 sampling techniques along with their parameters and implementation details. Second, we introduce the 18 datasets under study. Finally, we present the results. The code to run the experiment, the results, and a standalone version of our algorithms, are available online[1].

A. Implementation

We implemented six baseline techniques introduced in Section III, and our three proposed approaches introduced in Section V.

Some of these strategies are variant-based, so we apply the EOR to allow duplicate traces in the sampling. Table II provides an overview of the 9 techniques implemented. While we introduced them in their respective sections, we provide some implementation details when needed.

TABLE II
Overview of the compared techniques.

Section	Abbr.	Name	EOR
III-B1	T1	Random	
III-B2	T2	Variants stratified	✓
III-B3	T3	Variants biased	
III-B4	T4	Behaviour-based	✓
III-B5	T5	LogRank	✓
III-B6	T6	Redundancy Check	✓
V-A	T7	Iterative c-min	
V-B	T8	Iterative c-min (EOR)	✓
V-C	T9	Iterative c-min (Eucl.)	✓

T1 is drawn using random sampling with replacement. T2 is a random sampling leveraging the EOR, which acts equivalently to the stratified sampling introduced in Section III-B2. T3 is a biased sampling technique that is set to return the most frequent variants. T4 is a behaviour-based sampling that requires choosing which are the most prominent and rarest behaviours. We retrieve, respectively, the first and last tertiles of the behaviours. T5 leverages the page rank algorithm and requires a way to measure the distance between variants. We use n-grams of sizes one and two and the euclidean distance to reflect the original implementation [17]. Moreover, to sample under the 10 minutes threshold, we limited the maximum number of neighbourhoods to 200 using a K-Nearest Neighbour (KNN) algorithm. For T6, we measure the centrality using the cost matrix (of the normalized Levenshtein distances) and set the min distance for a variant to be considered to 0.05.

The last three implementations are our proposed approaches from Section V. T7 and T8 also use the cost matrix. T8 aims to reduce the execution time of T7 by applying the EOR first. Finally, T9 implements the iterative c-min that avoids the need for a cost matrix by working in Euclidean space. It uses n-grams of size one and two reduced to 64 dimensions through SVD and leverages an efficient implementation of the k-nearest neighbour algorithm, namely Faiss [26]. Note that we do not leverage the CPU or approximation capacities of the Faiss library, which could further reduce the execution time.

[1] https://github.com/gaelbernard/sampling

B. Datasets

We sampled 18 mainstream datasets from the process mining discipline to cover various log characteristics. We provide an overview of the datasets in Table III. In the same table, we report the percentage of traces covered by the 20 most frequent variants because we believe that it provides a good indication of the complexity linked to the sampling task (the less coverage, the more complex the task).

TABLE III
DATASETS OVERVIEW.

	dataset	#events	#trace	#variants	20cov*
1	BPI2011 [26]	150 291	1 143	981	14%
2	BPI2012 [27]	262 200	13 087	4 366	53%
3	BPI2013_1 [28]	2 351	819	182	74%
4	BPI2013_2 [28]	6 660	1 487	327	73%
5	BPI2013_3 [28]	65 533	7 554	2 278	54%
6	BPI2015_1 [29]	38 944	937	916	4%
7	BPI2015_2 [29]	33 373	645	644	3%
8	BPI2015_3 [29]	44 801	1 087	1 032	7%
9	BPI2015_4 [29]	34 848	787	781	3%
10	BPI2015_5 [29]	43 896	892	879	4%
11	BPI2017 [30]	1 160 405	31 509	14 972	23%
12	BPI2018 [31]	2 514 266	43 809	28 923	15%
13	BPI2020_1 [32]	36 796	6 886	89	97%
14	BPI2020_2 [32]	86 581	7 065	1 478	53%
15	BPI2020_3 [32]	18 246	2 099	202	82%
16	BPI2020_4 [32]	72 151	6 449	753	64%
17	BPI2020_5 [32]	56 437	10 500	99	98%
18	Sepsis [33]	15 190	1 049	845	17%

*percentage of traces covered by the 20 most frequent variants.

C. Results

In Fig. 2, we report the EMDs for the BPI2020_2 dataset. Our proposed approaches T7 and T8 return the subset of traces that minimize the EMD the most. Note that we cannot distinguish them in Fig. 2 they achieve the same EMD. Although we made this type of chart for the 18 datasets (available with the code), we cannot show them in this paper due to space constraints. We hence turn to alternatives representations.

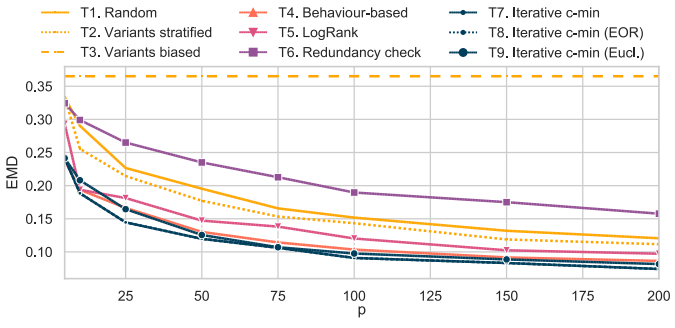

Fig. 2. Detailed results for the BPI2020_2 dataset.

In Fig. 3, we report the percentage change compared to the baseline T1. The first line of Fig. 3, T1, is not visible because it reports the difference with itself. As expected, the biased approach T3 performs poorly. Contrastingly, the iterative c-min samplers (T7, T8, and T9) stood out with a median

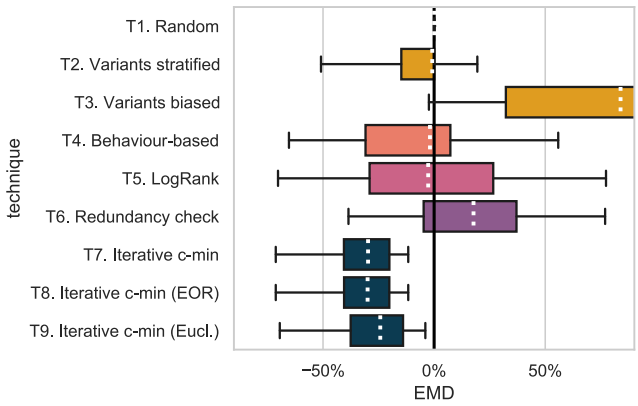

Fig. 3. EMD percentage change compared to T1.

reduction of a least 25% percentage change. Additionally, T7 and T8 perform slightly better than T9.

In Fig. 4, we report the EMD for 4 out of the 9 techniques to increase the readability. We choose to disregard five techniques for the following reasons. On average T1, T3, and T6 produce sampled event logs with higher EMD (visible in Fig. 3). Moreover, we do not display T7 and T8 because of the missing values due to getting over the 10 minutes threshold limit. In Fig. 4, it can be seen that the range of the EMD is dataset-dependent. For instance, an EMD below 0.35 for BPI2011 is good, while it would be a high EMD for BPI2020_1. The inherent complexity of the datasets explains the difference. We also note that T9 performs well consistently, while the other approaches fluctuate considerably by dataset. Also, the larger the p, the lower the EMD. Finally, BPI2020_1 and BPI2020_5 seem to be the only datasets where the EMD approaches zero. Looking back at Table III, it is not surprising since these datasets have a low number of variants, and 20 traces cover more than 97% of the traces.

Similar to how we displayed the EMD in Fig. 3, Fig. 5 reports the ESB (introduced in Section III-A). Although these metrics are different, the performance is comparable. Interestingly, T9 is slightly better compared to T7 and T8. Also, note that T4, designed to perform well on such behavioural metrics, has a more negligible improvement than the iterative c-min suites (T7, T8, T9).

We investigated further the correlation between the EMD and the ESB using Pearson's correlation tests. The correlations are reported below in APA style for each dataset:

1) $r(286) = .75, p < .01$
2) $r(281) = .83, p < .01$
3) $r(286) = .86, p < .01$
4) $r(286) = .76, p < .01$
5) $r(286) = .69, p < .01$
6) $r(286) = .74, p < .01$
7) $r(286) = .88, p < .01$
8) $r(286) = .76, p < .01$
9) $r(286) = .82, p < .01$
10) $r(286) = .82, p < .01$
11) $r(222) = .77, p < .01$
12) $r(217) = .62, p < .01$
13) $r(286) = .92, p < .01$
14) $r(286) = .88, p < .01$
15) $r(286) = .90, p < .01$
16) $r(286) = .90, p < .01$
17) $r(285) = .91, p < .01$
18) $r(286) = .81, p < .01$

By dataset, the maximum number of observations is 288 (8 p-values · 9 techniques · 4 repetitions), from which we

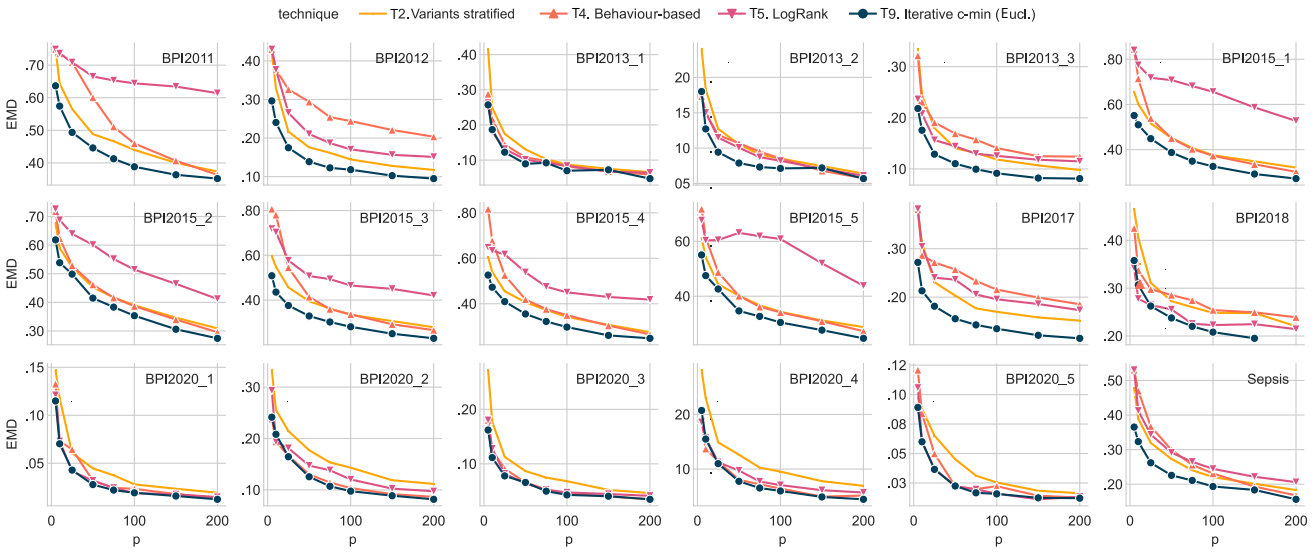

Fig. 4. EMD results. Only four techniques (T2, T4, T5, T9) are shown for purposes of readability.

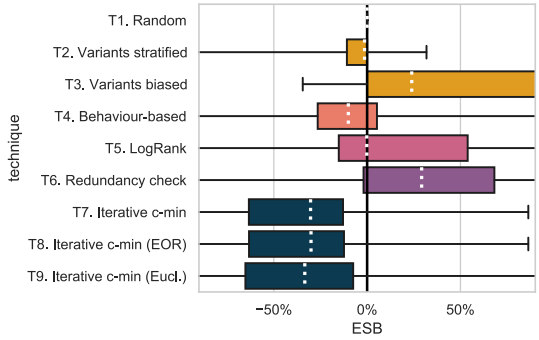

Fig. 5. Error sampling percentage change compared to T1.

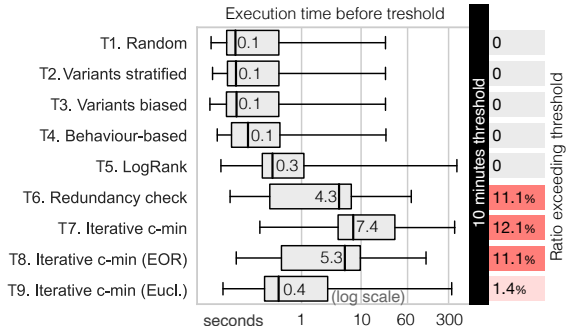

Fig. 6. Sampling execution time. The median is reported.

remove 2 to get the degree of freedom, i.e., the maximum degree of freedom is 286. However, the values reported do not always reach 286 because some experiments exceeded the 10 minutes threshold. As can be seen, the correlation is very strong (> 0.8) for 11 datasets and strong (> 0.6) for the 7 remaining ones. These high correlations highlight that these two metrics are closely related; i.e., improving one most likely improves the other as well.

We report the sampling execution time and the ratio of experiments that exceeded the 10 minutes duration threshold in Fig. 6, which can be summarized in three groups. First, T1, T2, T3, and T4 are the techniques that do not require the calculation of the distance between traces. They are hence the fastest ones with a median execution time below 0.1 seconds. Second, T5 and T9 work in Euclidean space and leverage feature reduction techniques (i.e., SVD) and advanced indexing techniques (i.e., KNN). The median execution time of these techniques is below 0.4 seconds. Third, T6, T7, and T8 spend a significant time building the cost matrix. Consequently, these techniques could not complete within the 10-minutes threshold in approximately 11% of the cases. Interestingly, the iterative c-min with EOR (T8) reduces the median execution time of T7 by approximately 28%–from 7.4s to 5.3s median time–without impacting the sampling quality (visible in Fig. 3 and Fig. 5). Hence, we suggest that T8 is always preferable over T7. However, for event logs composed of many variants, this would not prevent reaching the threshold since the EOR does not reduce the time needed to compute the cost matrix.

In the next section, we discuss these results as well as the limitation of the experiment.

VII. Discussion

We demonstrated that the EOR is an effective heuristic to reduce the computation time and allows turning a variant-based sampling into a trace-based one, i.e., allowing repetitions when a sampling technique is initially not designed to return them. However, when the number of unique variants exceeds few thousand, the time and space needed to build the cost matrix become prohibitive and applying the EOR does not solve the issue. Hence, as a rule of thumb, we suggest using sampling technique T8 when the number of unique variants is below one or two thousand and applying T9 for larger event

logs. Note that the execution time of T9 could be improved by reducing the number of dimensions or approximating the KNN search. Nevertheless, the impact of such optimization is out-of-scope of this paper.

Similarly, some metrics and algorithms require various parameters. Among others, we can mention the bandwidth of the ESB, the threshold for the redundancy check or the number of prominent are rare behaviours to consider in the behaviour-based sampling. We implemented the algorithms and metrics with due diligence as described by the authors and choose the set of parameters that we deemed appropriate for the task at hand. However, the search space of all possible parameters makes an exhaustive grid search inconceivable. Nonetheless, given the extensive number of experiments reported in this work, we are confident about the conclusion drawn in the experimental section.

VIII. Conclusion

We benchmarked several datasets (18), techniques (9), sampling sizes (10) and repeated the experiment multiple times (4). With the extensive set of experiments and the fact that we use two distinct evaluation criteria, we confidently claim that the proposed iterative c-min suite (T7, T8, and T9) discovers, in general, more representative sample event logs.

Downsizing event logs in a representative way can streamline the exploratory phase of process mining projects. In fact, not only can it help approximate time-consuming algorithms (e.g., conformance checking), but it can also present a fair summary to a process mining analyst by answering questions such as *"what are the 20 traces that best summarize the event logs?"* Thus, we believe that the sampling of event logs can play an important role in extracting insights from complex event logs.

References

[1] M. F. Sani, "Preprocessing event data in process mining." in *CAiSE (Doctoral Consortium)*, 2020, pp. 1–10.

[2] M. F. Sani, S. J. van Zelst, and W. M. van der Aalst, "The impact of biased sampling of event logs on the performance of process discovery," *Computing*, pp. 1–20, 2021.

[3] S. J. Leemans, W. M. van der Aalst, T. Brockhoff, and A. Polyvyanyy, "Stochastic process mining: Earth movers' stochastic conformance," *Information Systems*, p. 101724, 2021.

[4] B. Knols and J. M. E. van der Werf, "Measuring the behavioral quality of log sampling," in *2019 International Conference on Process Mining (ICPM)*. IEEE, 2019, pp. 97–104.

[5] W. M. van der Aalst, *Process Mining: Data Science in Action*. Berlin, Heidelberg: Springer, 2016. [Online]. Available: https://doi.org/10.1007/978-3-662-49851-4_1

[6] W. Van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE transactions on knowledge and data engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.

[7] S. J. Leemans, D. Fahland, and W. M. van der Aalst, "Discovering block-structured process models from event logs-a constructive approach," in *International conference on applications and theory of Petri nets and concurrency*. Springer, 2013, pp. 311–329.

[8] R. Conforti, M. La Rosa, and A. H. ter Hofstede, "Filtering out infrequent behavior from business process event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 2, pp. 300–314, 2016.

[9] M. F. Sani, S. J. van Zelst, and W. M. van der Aalst, "Improving process discovery results by filtering outliers using conditional behavioural probabilities," in *International Conference on Business Process Management*. Springer, 2017, pp. 216–229.

[10] M. Bauer, H. van der Aa, and M. Weidlich, "Estimating process conformance by trace sampling and result approximation," in *International Conference on Business Process Management*. Springer, 2019, pp. 179–197.

[11] M. F. Sani, S. J. van Zelst, and W. M. van der Aalst, "Conformance checking approximation using subset selection and edit distance," in *International Conference on Advanced Information Systems Engineering*. Springer, 2020, pp. 234–251.

[12] M. Bauer, A. Senderovich, A. Gal, L. Grunske, and M. Weidlich, "How much event data is enough? a statistical framework for process discovery," in *International Conference on Advanced Information Systems Engineering*. Springer, 2018, pp. 239–256.

[13] A. Berti, "Statistical sampling in process mining discovery," in *The 9th International Conference on Information, Process, and Knowledge Management*, 2017, pp. 41–43.

[14] J. M. E. van der Werf, A. Polyvyanyy, B. R. van Wensveen, M. Brinkhuis, and H. A. Reijers, "All that glitters is not gold: Towards process discovery techniques with guarantees," *arXiv preprint arXiv:2012.12764*, 2020.

[15] N. Busany and S. Maoz, "Behavioral log analysis with statistical guarantees," in *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. IEEE, 2016, pp. 877–887.

[16] A. Gabadinho, G. Ritschard, M. Studer, and N. S. Müller, "Extracting and rendering representative sequences," in *3rd International Joint Conference on Knowledge Discovery, Knowlege Engineering and Knowledge Management (IC3K)*. Berlin, Heidelberg: Springer, 2011, pp. 94–106.

[17] C. Liu, Y. Pei, L. Cheng, Q. Zeng, and H. Duan, "Sampling business process event logs using graph-based ranking model," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, p. e5974, 2021.

[18] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[19] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8. Soviet Union, 1966, pp. 707–710.

[20] S. J. Leemans, A. F. Syring, and W. M. van der Aalst, "Earth movers' stochastic conformance checking," in *International Conference on Business Process Management*. Springer, 2019, pp. 127–143.

[21] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 59–66.

[22] O. Pele and M. Werman, "Fast and robust earth mover's distances," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 460–467.

[23] S. L. Hakimi, "Optimum distribution of switching centers in a communication network and some related graph theoretic problems," *Operations research*, vol. 13, no. 3, pp. 462–475, 1965.

[24] V. Maniezzo, A. Mingozzi, and R. Baldacci, "A bionomic approach to the capacitated p-median problem," *Journal of Heuristics*, vol. 4, no. 3, pp. 263–280, 1998.

[25] H. R. Lewis, "Computers and intractability. a guide to the theory of np-completeness," 1983.

[26] B. van Dongen, "Real-life event logs - Hospital log," 2011, doi:10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54.

[27] ——, "BPI Challenge 2012," 2012, doi:10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.

[28] W. Steeman, "BPI Challenge 2013," 2013, doi:10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07, (1) open problem, (2) closed problem, (3) incidents.

[29] J. Buijs, "Environmental permit application process ('WABO'), CoSeLoG project – Municipality 1-5," 2014, doi:10.4121/uuid:31a308ef-c844-48da-948c-305d167a0ec1 (1) municipality 1, (2) municipality 2, (3) municipality 3, (4) municipality 4, (5) municipality 5.

[30] B. van Dongen, "BPI Challenge 2017," 2017, doi:10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b.

[31] B. van Dongen and F. F. Borchert, "BPI Challenge 2018," 2018, doi:10.4121/uuid:3301445f-95e8-4ff0-98a4-901f1f204972.

[32] B. van Dongen, "BPI Challenge 2020," 2020, doi:10.4121/uuid:52fb97d4-4588-43c9-9d04-3604d4613b51 (1) request for payment, (2) permit log, (3) prepaid travel cost, (4) international declarations, (5) domestic declarations.

[33] F. Mannhardt, "Sepsis Cases - Event Log," 2016, doi:10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460.

View publication stats