AVOCADO: The Streaming Process Mining Challenge

Christian Imenkamp [1], Andrea Maldonado [2], Hendrik Reiter [3], Martin Werner [2], Wilhelm Hasselbring [3], Agnes Koschmider [1], and Andrea Burattin [4]

Abstract: Streaming process mining deals with the real-time analysis of streaming data. Event streams require algorithms capable of processing data incrementally. To systematically address the complexities of this domain, we propose AVOCADO, a standardized challenge framework that provides clear structural divisions: separating the concept and instantiation layers of challenges in streaming process mining for algorithm evaluation. The AVOCADO evaluates algorithms on streaming-specific metrics like accuracy, Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Processing Latency, and robustness. This initiative seeks to foster innovation and community-driven discussions to advance the field of streaming process mining. We present this framework as a foundation and invite the community to contribute to its evolution by suggesting new challenges, such as integrating metrics for system throughput and memory consumption, and expanding the scope to address real-world stream complexities like out-of-order event arrival.

Keywords: Streaming Process Mining, Streaming Conformance, Concept Drifts, Streaming Challenge

1 Introduction

Traditionally, process mining focuses on process discovery, the task of extracting process models from event data. These event logs, typically represent finite sets of recorded activities, often used to model the flow of a business process. While such logs provide a forensical view of past executions, their static and bounded nature limits the ability of process mining techniques to generalize beyond historical cases [Fi22]. In recent years, the need to extend process mining techniques to handle event streams has grown and thus to efficiently handle continuous, potentially infinite sequences of events. Unlike static logs, stream-based analysis enables the derivation of immediate insights into ongoing process executions. This allows organizations to monitor, adapt, and optimize their processes as they occur.

The shift from event logs to event streams in process mining introduces several challenges that challenge process mining. While event logs consist of finite, static data representing

[1] University of Bayreuth, Bayreuth, Germany,
christian.imenkamp@uni-bayreuth.de, https://orcid.org/0009-0007-4295-1268;
agnes.koschmider@uni-bayreuth.de, https://orcid.org/0000-0001-8206-7636

[2] School of Engineering and Design, Technical University of Munich, Germany,
andrea.maldonado@tum.de, https://orcid.org/0009-0009-8978-502X;
martin.werner@tum.de, https://orcid.org/0000-0002-6951-8022

[3] Christian-Albrechts-University Kiel, Kiel, Germany,
hendrik.reiter@email.uni-kiel.de, https://orcid.org/0009-0003-8544-0012;
hasselbring@email.uni-kiel.de, https://orcid.org/0000-0001-6625-4335

[4] DTU Compute, Technical University of Denmark, Kongens Lyngby, Denmark,
andbur@dtu.dk, https://orcid.org/0000-0002-0837-0183

past activities, event streams are continuous and real-time, requiring algorithms to process data as it arrives. This real-time processing demands algorithms that can incrementally update process models without access to the entire dataset at once, placing constraints on both memory and computational resources. Beyond the new challenges, the real-time environment also introduces new opportunities for process mining. In offline processing, the discovery of process models from complete historical data is central. Conversely, real-time environments can emphasize tasks more suited to immediate actionability. In particular, in real-time settings, the continuous update of process models provides limited practical value for most practitioners. They primarily seek actionable insights rather than observing model evolutions [Am25]. Recent empirical evidence shows that users interact with process mining mainly through dashboards and performance indicators. Conversely, streaming conformance checking becomes significantly more valuable in real-time scenarios. A low conformance score indicates an immediate deviation from expected behavior during the current process execution.

To address these challenges, we propose AVOCADO, which is designed to evaluate process mining algorithms specifically for event streams. In particular, AVOCADO will focus on streaming conformance algorithms. It provides a standardized framework for assessing algorithms on their ability to process continuous event data while balancing accuracy and performance under drifting real-time conditions. By using synthetic event data generated from complex process models, AVOCADO offers clear, objectified evaluation criteria that reflect the challenges inherent in working with event streams. This challenge advances the state of process mining by driving the development of algorithms that can learn the expected behavior from event streams, while meeting the system and resource constraints typical of real-time, large-scale environments. Ultimately, AVOCADO aims to foster process mining techniques allowing organizations to gain insights into their business processes in real-time, improving decision-making and operational efficiency.

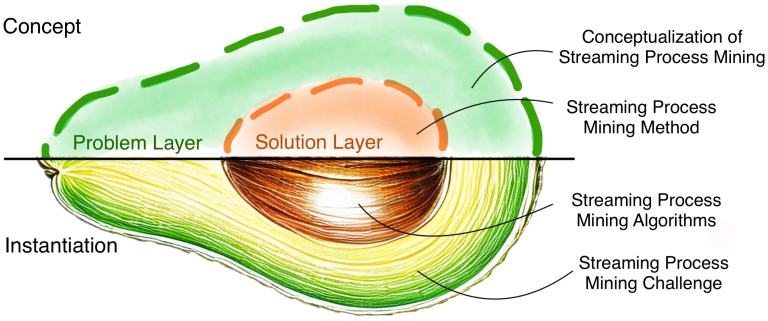

Fig. 1: AVOCADO: Unveiling the problem's 'pulp' to expose the solution set 'seed', conceived in the formalization of SPM and embodied in this challenge.

The conceptual framework guiding our work is visually depicted in Figure 1. This AVOCADO representation structures the Streaming Process Mining domain into distinct layers. At the conceptual level, it differentiates between the *problem layer* and the *solution layer*.

The instantiation of these concepts progresses from *Streaming Process Mining Methods* to *Streaming Process Mining Algorithms*, and the *Conceptualization of Streaming Process Mining* and culminating in our *Streaming Process Mining Challenge*. This layered model illustrates how the proposed challenge serves as a concrete instantiation for evaluating algorithms within the broader context of streaming process mining.

2 Related Work

This section discusses related work in terms of our addressed problem (see Sect. 2.1) and the related literature to our solution (see Sect. 2.2).

2.1 Problem Layer

2.1.1 Process Discovery Contest

The Process Discovery Contest (PDC), hosted annually at the International Conference on Process Mining (ICPM), serves as a prominent platform for evaluating the effectiveness and efficiency of process discovery algorithms. Historically, the contest has been structured around the offline batch processing of static event logs [Ca17]. The traditional PDC format, as outlined by [Ca17] and the ICPM 2025 contest descriptions[5], involves:

- **Static Event Logs**: Participants receive pre-generated, finite event logs (e.g., XES files) as input, clearly indicating the beginning and end of the data available for model discovery.
- **Batch Processing**: Algorithms are expected to process the entire log to produce a single, complete process model (e.g., in PNML or BPMN format).
- **Offline Evaluation**: Evaluation metrics such as fitness, precision, generalization, and simplicity are computed on the discovered model against a "ground truth"model derived from the static log. This process does not impose real-time constraints on continuous updates or performance.
- **Focus on Overfitting/Underfitting**: The primary objective is to achieve an optimal balance between overfitting (generating overly restrictive models) and underfitting (producing overly general models) to the provided log.

The traditional focus of process mining lies in analyzing historical static event logs for process discovery, monitoring, and improvement. However, the increasing availability of real-time event data from various IT systems has encouraged complementary research in streaming process mining. This emerging field addresses the unique challenges posed by unbounded,

[5] ICPM Process Discovery Contest 2025: https://icpmconference.org/2025/process-discovery-contest/

continuous streams of events, which are not present in conventional offline analysis. The fundamental objective of streaming process discovery is to incrementally generate a sequence of process models, where each model is intended to reflect process behavior observed within a recent window of events, drawn from an infinite event stream. This inherent need for continuous model generation and adaptation fundamentally distinguishes streaming process mining from its offline counterparts, introducing the following requirements for a challenge [Bu18a; ZDA18].

- **Evaluation in Incomplete Information**: Streaming event logs are inherently incomplete, as traces are ongoing and future behavior is unknown, complicating discovery. Furthermore, they often contain noisy data due to errors or system glitches, which algorithms must robustly manage alongside concept drift, where process behavior changes over time [BV20]. Additionally, processing events incrementally, often one by one, imposes strict resource constraints and unawareness of future events [ZDA18].
- **Handling of Concept Drifts**: The underlying process may evolve over time. This means that the mined models must adapt to such changes, balancing the ingestion of new behavior to avoid quick fluctuations due to short-term changes [Ze21]. Four distinct drift types are discussed: sudden (i.e., an instantaneous switch between models), gradual (i.e., probabilistic alternation between coexisting models), incremental (i.e., progressive transformation through intermediate variants), and recurring (i.e., reappearance of a previously active concept) [AS22].
- **Online Evaluation**: Unlike batch-based evaluation using complete logs for a single final model, online evaluation in streaming process mining demands metrics for a sequence of evolving models. This is compounded by inherently incomplete trace information, complicating discovery and quality assessment of dynamic process behavior. [Bu18a]
- **Significance of Ground Truth in an Online Scenario**: In an online scenario, the concept of a static "ground truth"model for evaluation becomes ambiguous due to continuous process evolution (concept drift). Evaluation must thus assess how well models generalize from partial and dynamic behavior, potentially against an evolving reference or based on real-time deviation detection.

2.2 Solution Layer

Various research efforts have addressed streaming process mining, each with distinct capabilities and limitations, highlighting open research avenues.

Algorithms for Concept Drift and Non-Stationary Processes. Batyuk and Voityshyn [BV18; BV20] proposed streaming discovery methods within Lambda architectures. In particular, it emphasizes operational support and recent event prioritization, but lacks explicit drift detection or noise management. Potoniec et al. [PSP22] developed an Online

Miner for rapid drift response in causal nets, without fully addressing noise resilience or efficiency. Zellner et al. [Ze21] improved drift detection precision via dynamic outlier aggregation but offered limited adaptive modeling. Colombo Tosatto et al. [Co25] tackled out-of-order events in compliance monitoring through reliable prefix identification and confidence scores, critical for robust analyses. While effective individually, these approaches often isolate drift and noise handling from core discovery processes.

Declarative and Low-Level Event Data Approaches. Burattin et al. [Bu15; BLS23] explored declarative online discovery and change detection using DCR graphs, yet neglected computational efficiency and explicit noise strategies. Lefebure et al. [LTC23] focused on deriving high-level attributes from technical event streams to handle frequency and asynchrony, though impacts on model quality remain unclear. These approaches broaden streaming discovery scope but incompletely address practical implementation challenges.

Online Conformance Checking. Online conformance checking methods provide real-time process assessments. Van Zelst et al. [Ze19] introduced incremental prefix-alignments balancing memory and approximation accuracy. Burattin et al. [Bu18b] presented an in-vivo deviation detection framework without predefined case start points. Nagy and Werner-Stark [NW22] extended alignment techniques with data perspectives for incomplete executions. Lee et al. [Le21] developed an HMM-based approach ensuring constant-time event orientation. Practical implementations, such as CCaaS [Wend], demonstrate operational viability. Yet, comprehensive metrics and optimal resource-accuracy trade-offs remain open challenges.

Despite advances, current streaming process discovery approaches primarily address individual challenges like scalability or basic drift handling in isolation. Integrative solutions combining adaptive modeling, resource efficiency, diverse drift management, and resilience to incomplete or noisy data remain crucial [IK25].

3 Conceptualization of Streaming Process Mining

In this section, we formalize the foundation of streaming process mining that underlies the AVOCADO framework. We outline how continuous event streams can be processed and evaluated in real time. In particular, we focusing on how algorithms operate under streaming-specific conditions.

Formally, let $S = \langle \dots, e_1, e_2, e_3, \dots \rangle$ denote an infinite stream of events, where each $e_i = (a, c)$ contains at least an activity label a and a case id c – also accessible via projection operators, i.e., $\pi_a(e_i) = a$ and $\pi_c(e_i) = c$. Then the goal of streaming process discovery is to incrementally generate a sequence of process models $\langle M_1, M_2, \dots \rangle$ such that each model M_t, discovered after the event with index t, reflects the behavior observed in the stream S over the latest δ events, i.e., $S_{[t-\delta,t]}$.

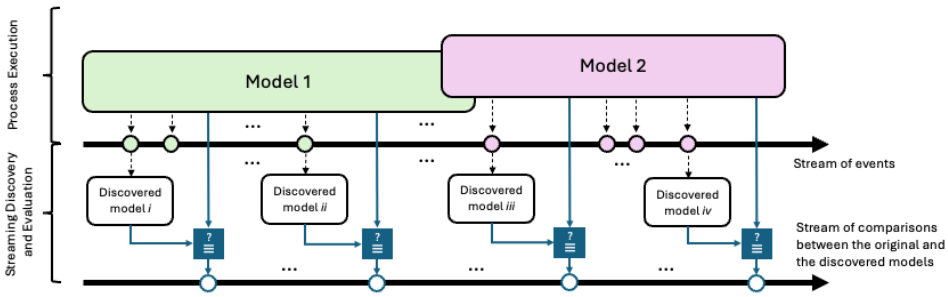

Fig. 2: Conceptualization of the "rediscovery problem" in the context of Streaming Process Mining.

When compared to traditional process mining, streaming process mining introduces several novel challenges, including:

- Evaluation on incomplete information: At any point in time, only a prefix or a subsequence of a trace may be available; thus, the discovery as well as the corresponding quality assessments are complicated.
- Concept drift: The underlying process may evolve over time. This means that the mined models must adapt to such changes, balancing the ingestion of new behavior to avoid quick fluctuations due to short-term changes.
- Resources and latency: Algorithms must operate with bounded memory and time performance constraints, limiting their ability to store full trace history. Additionally, the ability to generate updated process models should introduce the least possible delay to ensure the readiness of results.
- Noisy data: While incompleteness of data is an intrinsic challenge of streaming process mining, as in the offline case, noise may or may not represent one additional aspect that algorithms must handle.

In traditional process mining, the "rediscovery problem" [LFA14] refers to whether a control-flow discovery algorithm can discover a model M that exactly reflects the behavior observed in a log L from which it was derived. In the streaming context, considering the challenges mentioned above, this becomes more complex. The objective shifts from whether the evolving model correctly generalizes from partial and evolving behavior. Furthermore, unlike incremental process mining [Ca10], which updates models from growing logs, streaming process mining must update the model often, discarding earlier data due to memory constraints and changes becoming more relevant.

A graphical representation of the "rediscovery problem" is shown in Figure 2. Here, there are two models generating events. These two models follow one another with only a partial overlap. The events being generated into the "stream of events" refer to either one or the other. Each event is mined, and a corresponding model is discovered. Then, each discovered

model must be compared to the original model from which the event was generated. In the picture, the result of such a comparison generates a stream of comparison values.

As we aim for a modeling language-agnostic approach, the conceptualization of our rediscovery problem should not be bound to any fixed formalism (e.g., Petri net). Therefore, to accomplish this, rather than comparing the mined models with the original one, we decided to compute the conformance measure between the mined models and a log comprising the behavior of the reference model. It becomes clear that this evaluation simultaneously addresses two intertwined dimensions: the quality of the discovered model and the reliability of the conformance assessment itself. However, in the streaming context, conformance is not a straightforward measure, and it must be interpreted alongside other critical dimensions such as *confidence* and *completeness*. This, however, enables us to adopt a *language-agnostic* approach.

Furthermore, we address the internal procedure and process of the challenge and the tests. For this, we provide Algorithm 1: First (lines 1-4), two process models (i.e., p and k) are constructed. While p is fully random, k contains at least 85% of the same activities as k ($k.a$ indicates the set of activities of k). Furthermore, p and k have entirely distinct directly-follow relations ($k.df$ indicates the set of directly-follow relations of k), i.e., k is a reordering of p. The union of p and k is called w (i.e., a driving log). While S denotes the stream in general, S_{train} can be used by the algorithms to learn the behavior of the stream. Furthermore, S_{val} is the subset of the stream to evaluate the algorithm. S_{train} is constructed using a play-out of p, and is used for the warm-up phase where the algorithms can learn the behavior of the stream (lines 7-9). Next (lines 5-6), is the generation of the event stream for validation by construction of a drifting stream with $p \rightarrow w \rightarrow k$. Then, the ground truth (gt) is constructed: Let denote $e \in S_{val}$ and $gt(e)$ the ground truth conformance of event e:

$$gt(e) = \sum_{e \in S_{\text{val}}} \begin{cases} 1 & \text{if } e \in p \\ 0.5 & \text{if } e \in w \\ 0 & \text{if } e \in k \end{cases}$$

After that (line 10-13), is the main challenge, where the participant's algorithm gets each event using the conformance method. The calculated conformance (i.e., the user prediction if the event is from p or k) is stored and compared against the ground truth (line 14). The global evaluation score E_{global} is computed as the mean deviation between the predicted and ground-truth values over all validation events.

As outlined in Sect. 4.1, concept drifts represent a fundamental challenge and thus a key requirement.

Therefore, we demonstrate how the combination of process models p and k can reproduce all principal drift types

Algorithm 1 Algorithm Evaluation Procedure

Require: $\textsc{GenerateStream}(m, n)$, $\textsc{GetRandomProcess}()$, algorithm a implementing $\textsc{learn}(e)$ and $\textsc{conformance}(e)$, error function $\textsc{compare}(x, y)$
Ensure: E_{global}: average deviation between ground truth and predictions

```
p ← GetRandomProcess()
k ← GetRandomProcess() such that overlap(k.a, p.a) ≥ 85% and k.df ≠ p.df
w ← p ∪ k
Strain ← GenerateStream(p)
Sval ← GenerateStream(p, 100) || GenerateStream(w, 50) || GenerateStream(k, 100)
gt ← [1^100, 0.5^50, 0^100]
for each event e in Strain do
    a.learn(e)
end for
c ← [ ]
for i = 1 to |Sval| do
    e ← Sval[i]
    score ← a.conformance(e)
    append score to c
end for
E_global ← (1/|c|) Σ_{i=1}^{|c|} compare(gt[i], c[i])
return E_global
```

A sudden drift occurs when the concatenation $S_{\text{val}} = \text{GenerateStream}(p) \parallel \text{GenerateStream}(k)$ yields an immediate model change $p \rightarrow k$. A gradual drift emerges when the inclusion of w in $S_{\text{val}} = p \parallel w \parallel k$ enables alternating behavior between p and k within a bounded interval, leading to probabilistic coexistence. An incremental drift results from gradually adjusting the relative frequency of events from p and k inside w. This produces intermediate hybrids between both models and forming a continuous transition path. Finally, a recurring drift arises since w preserves both behavioral patterns, and repeating the sequence $p \rightarrow k \rightarrow p$ reproduces previously active process structures.

4 Streaming Process Mining Challenge

We now turn the discussion to AVOCADO. Our repository is publicly available[6] and the challenge can be accessed online.[7] First, we address the requirements and general design of the algorithms in Sect. 4.1. Next, in Sect. 4.2, we present the underlying pipeline and the steps required to participate in the challenge. Finally, we discuss the parameters, metrics, and the structure of the event stream in Sect. 4.3.

[6] https://github.com/chimenkamp/Streaming-Process-Mining-Challenge
[7] https://streaming-process-mining-challenge.onrender.com/

4.1 Algorithm Requirements

Participants must adhere to a few general guidelines when developing the conformance algorithm. This is necessary to ensure comparability, reproducibility, and successful execution during the challenge.

First, algorithms must include a main class that implements the provided interface (i.e., *BaseAlgorithm*). This is necessary to ensure that the required methods are implemented and aligned with the predefined structure. The interface includes two methods (*learn* and *conformance*). The *learn* method is responsible for the **Warm-Up** phase, where the algorithm has the opportunity to learn the behavior of the stream. The *conformance* method is called with each new event and should return the current conformance calculated by the algorithm.

Second, participants must include a clear and executable start script located in the root of the directory. This script should contain the main class that implements the interface and all the necessary configurations. Additionally, submissions must not contain any top-level code that is executed by just loading the script (e.g., `if __name__ == "__main__"`). Moreover, algorithms must be capable of running solely on the CPU. We are explicitly excluding GPU support for the challenge. Although participants may utilize machine learning techniques, any required computation must occur within the CPU.

Finally, participants must explicitly specify any external Python dependencies in a *requirements.txt* file located in the root directory. Dependencies will be installed using the standard Python package manager (PIP) prior to the algorithm evaluation.

4.2 Participation

The challenge itself is web-based and can be accessed online[8]. To simplify participation in the challenge, we now address the necessary steps to submit an algorithm (see Fig. 3). The participation steps are detailed below:

Implement Your Algorithm. The implementation of the algorithm should adhere to the requirements outlined in Sect. 4.1

Understand Event Structure. Events are provided as dictionaries containing standard fields: `case:concept:name` (case identifier), `concept:name` (activity name), `time:timestamp` (event timestamp). Furthermore, in test streams, an additional field `concept:origin` (origin log for concept drifts) is provided.

Upload Your Algorithm. Participants navigate to the **Test & Submit** tab in the user interface. It is possible to upload a single Python file or a ZIP archive containing all required

[8] https://streaming-process-mining-challenge.onrender.com/

files (including the requirements.txt). The platform will automatically detect the class that implements the interface.

Configure and Run Test. After the upload, participants can run tests before submitting to the challenge. In particular, they can select different test streams and review the results. The results contain a conformance plot, performance metrics, and a baseline comparison (see Sect. 4.3 for more details).

Submit To Challenge. Post-testing participants submit their algorithm using the **Submit to Challenge** button. They must fill out a submission form, including team name, contract email, algorithm name, and a brief description of their algorithm.

Submissions' statuses and histories can be reviewed under the **My Submissions** tab. Additionally, the **Leaderboard** tab provides real-time rankings and performance comparisons among all participating teams.

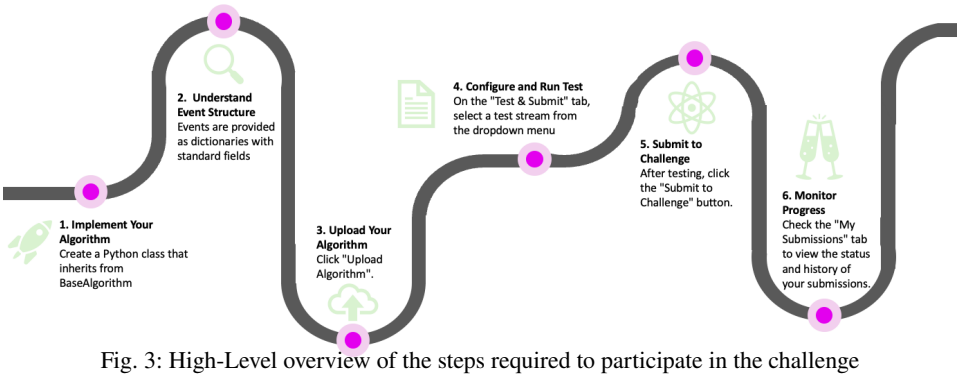

Fig. 3: High-Level overview of the steps required to participate in the challenge

4.3 Parameters and Metrics

Following, we address the evaluation metrics and parameters provided by the challenge:

Accuracy The percentage of predictions within 10% of the ground truth conformance score [ST15; Al18].

Mean Absolute Error (MAE) Average absolute distance between the algorithm conformance scores and the baseline scores [ST15; Al18].

Root Mean Square Error (RMSE) Squared root of the mean squared difference between the algorithm and the ground truth conformance scores [ST15; Al18].

Processing Latency Normalized metric based on average processing time per event, favoring lower times [ST15; Al18].

Robustness Score Reflects the number of major errors (difference greater than 30% from the ground truth) [ST15; Al18].

These metrics are combined into the score displayed on the leaderboard. The *score* is calculated by:

$$\begin{aligned} score = &(0.3 \times accuracy) \\ &+ (0.25 \times MAE) \\ &+ (0.2 \times RMSE) \\ &+ (0.15 \times latency) \\ &+ (0.1 \times robustness) \end{aligned}$$

The corresponding weights are open for discussion and may be adjusted on the basis of feedback from the community.

Furthermore, the platform includes several parameters for testing purposes. Participants can configure: **Test Stream Selection**: Choose from predefined streams (e.g., "Sudden", "Gradual", "Incremental", "Recurring"), each described by drift characteristics and baseline behavior. **Number of Cases**: Specify the event stream size by cases.

5 Assessment of Requirements

We evaluate how the proposed AVOCADO addresses the requirements identified for streaming process mining (see Sect. 2.1). In particular, we introduce a simple baseline algorithm as a test case to assess the requirements. It is called the *Frequency-Based Conformance Algorithm* and learns frequencies of activities and directly follows relations during the learning phase. More frequent activities or transitions are considered to be more conformant. Fig. 4 shows the web interface on the *Test & Submit* page.

Evaluation in Incomplete Information The AVOCADO addresses this by requiring algorithms to incrementally update their results when a new event arrives at the system. The stream is non-deterministic, and therefore future events are impossible to know. Thus, the challenge explicitly evaluates the robustness of the model to incomplete traces.

Handling of Concept Drifts The generation of event streams directly addresses this requirement (see Algorithm 1). In particular, by constructing validation streams with controlled concept drift scenarios (i.e., sudden, recurring, gradual, incremental) the challenge allows a detailed assessment of algorithms. Furthermore, algorithms are assessed on their ability to learn the behavior (i.e., conformance) of the stream and are evaluated based on the expected drop in conformance after the stream. Hence, algorithms aren't expected to show robustness against concept drifts.

Online Evaluation AVOCADO implements an online evaluation framework. For that, it continuously evaluates algorithms with metrics that reflect their real-time performance.

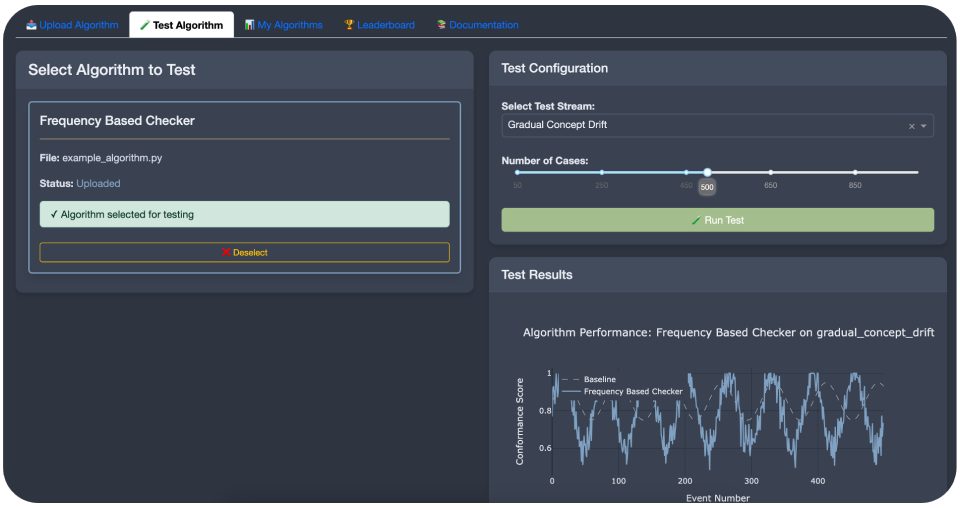

Fig. 4: Case-Study to assess the requirements of AVOCADO based on the frequency-based conformance algorithm

The adaptability and accuracy are continuously monitored using metrics such as accuracy, MAE, RMSE, and robustness. This ensures a realistic evaluation of algorithm performance in dynamic environments.

Significance of Ground Truth in an Online Scenario In streaming contexts, defining and maintaining ground truth is challenging. AVOCADO addresses this by reducing the complexity and defining the simplest possible ground truth. If the new event is from the process before the drift, we expect a conformance of one after a conformance of zero.

This AVOCADO incorporates adaptive validation because the framework continuously re-assesses algorithm performance during live event ingestion. Each algorithm dynamically adapts its validation to the stream conditions. This reflects the dynamic outcomes that evolve with drift and noise.

6 Conclusion

In this paper, we have addressed the emerging challenges of streaming process mining. In particular, the relevance of conformance in real-time environments. We have outlined the significant differences and new quality dimensions introduced by streaming data (i.e., including handling incomplete information, concept drift, real-time evaluation, and ambiguity of ground truth). To foster advancements in this area, we proposed the AVOCADO, based on the process discovery contest, offering a standardized web-based platform to

assess algorithms' performance. For that, we consider metrics like accuracy, MAE, RMSE, Processing Latency, and robustness.

This paper, however, does not propose a definitive or prescriptive solution. Instead, it is intended as an invitation for discourse and collaboration, specially about parameters, evaluation metrics, and their weights in our score. Recognizing the complexity and novelty of streaming process mining, our proposed challenge framework and evaluation criteria are meant as a foundation for further discussion. We acknowledge that our approach and selected metrics might not capture all nuances relevant for streaming settings. Thus, we explicitly invite researchers and practitioners to critique, refine, and expand upon our concepts and evaluation methods. By embracing a community-based approach, we believe we can collectively establish a more robust, comprehensive understanding of streaming process mining and continuously improve the challenge design.

Thus, AVOCADO provides not only a static benchmark but an adaptive validation mechanism for continuous performance assessment in streaming environments.

References

[Ca10] Cattafi, M. et al.: Incremental Declarative Process Mining. In (Szczerbicki, E.; Nguyen, N. T., eds.): Smart Information and Knowledge Management: Advances, Challenges, and Critical Issues. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 103–127, 2010, https://doi.org/10.1007/978-3-642-04584-4_5.

[LFA14] Leemans, S. J. J.; Fahland, D.; van der Aalst, W. M. P.: Discovering Block-Structured Process Models from Incomplete Event Logs. In (Ciardo, G.; Kindler, E., eds.): Application and Theory of Petri Nets and Concurrency. Springer International Publishing, Cham, pp. 91–110, 2014.

[Bu15] Burattin, A. et al.: Online Discovery of Declarative Process Models from Event Streams. IEEE Transactions on Services Computing 8 (6), Conference Name: IEEE Transactions on Services Computing, pp. 833–846, 2015, http://ieeexplore.ieee.org/document/7164336/, visited on: 11/09/2023.

[ST15] Singh, S. A.; Tirthapura, S.: An Evaluation of Streaming Algorithms for Distinct Counting over a Sliding Window. Frontiers in ICT 2, Section: Big Data Networks. Edited by Graham Cormode. Reviewed by Justin Thaler and Ahmed Metwally. Received: 20 August 2015; Accepted: 19 October 2015; Published: 05 November 2015. Original Research Article., p. 23, 2015, https://doi.org/10.3389/fict.2015.00023.

[Ca17] Carmona, J. et al.: Summary of the process discovery contest 2016. In. Springer International Publishing, 2017.

[Al18] Alipourfard, O. et al.: A Comparison of Performance and Accuracy of Measurement Algorithms in Software. In: Proceedings of the Symposium on SDN Research. SOSR '18, Association for Computing Machinery, Los Angeles, CA, USA, 2018, https://doi.org/10.1145/3185467.3185475.

[BV18] Batyuk, A.; Voityshyn, V.: Streaming Process Discovery for Lambda Architecture-Based Process Monitoring Platform. In. Vol. 1, pp. 298–301, 2018, visited on: 02/24/2025.

[Bu18a] Burattin, A.: Streaming Process Discovery and Conformance Checking. In (Sakr, S.; Zomaya, A., eds.): Encyclopedia of Big Data Technologies. Springer International Publishing, Cham, pp. 1–8, 2018, https://doi.org/10.1007/978-3-319-63962-8_103-1, visited on: 02/22/2025.

[Bu18b] Burattin, A. et al.: Online Conformance Checking Using Behavioural Patterns. In: Business Process Management. Springer International Publishing, pp. 250–267, 2018.

[ZDA18] van Zelst, S. J.; van Dongen, B. F.; van der Aalst, W. M. P.: Event stream-based process discovery using abstract representations. en, Knowledge and Information Systems 54 (2), pp. 407–435, 2018, visited on: 02/24/2025.

[Ze19] van Zelst, S. J. et al.: Online conformance checking: relating event streams to process models using prefix-alignments. International Journal of Data Science and Analytics 8 (3), pp. 269–284, 2019.

[BV20] Batyuk, A.; Voityshyn, V.: Streaming Process Discovery Method for Semi-Structured Business Processes. In. Pp. 444–448, 2020, https://ieeexplore.ieee.org/abstract/document/9204201, visited on: 02/24/2025.

[Le21] Lee, W. L. J. et al.: Orientation and conformance: A HMM-based approach to online conformance checking. Information Systems 102, p. 101674, 2021.

[Ze21] Zellner, L. et al.: Concept Drift Detection on Streaming Data with Dynamic Outlier Aggregation. In (Leemans, S.; Leopold, H., eds.): Process Mining Workshops. Springer International Publishing, Cham, pp. 206–217, 2021.

[AS22] Agrahari, S.; Singh, A. K.: Concept Drift Detection in Data Stream Mining : A literature review. Journal of King Saud University - Computer and Information Sciences 34 (10, Part B), pp. 9523–9540, 2022, https://www.sciencedirect.com/science/article/pii/S1319157821003062.

[Fi22] Fischer, D. et al.: Towards interactive event log forensics: Detecting and quantifying timestamp imperfections. Information Systems 109, p. 102039, 2022, https://www.sciencedirect.com/science/article/pii/S0306437922000382.

[NW22] Nagy, Z.; Werner-Stark, A.: An Alignment-based Multi-Perspective Online Conformance Checking Technique. Acta Polytechnica Hungarica 19 (4), pp. 105–127, 2022.

[PSP22] Potoniec, J.; Sroka, D.; Pawlak, T. P.: Continuous discovery of Causal nets for non-stationary business processes using the Online Miner. European Journal of Operational Research 303 (3), pp. 1304–1320, 2022, visited on: 02/24/2025.

[BLS23] Burattin, A.; López, H. A.; Starklit, L.: Uncovering Change: A Streaming Approach for Declarative Processes: 4th International Conference on Process Mining. In (Montali, M.; Senderovich, A.; Weidlich, M., eds.): Process Mining Workshops. Vol. 468, Publisher: Springer, Springer Nature Switzerland, Cham, pp. 158–170, 2023.

[LTC23] Lefebure, F.; Thuault, C.; Cholet, S.: Real-World Streaming Process Discovery from Low-Level Event Data. In (Sellmann, M.; Tierney, K., eds.): Learning and Intelligent Optimization. Springer International Publishing, Cham, pp. 150–164, 2023.

[Am25] Ammann, J. et al.: How do process mining users act, think, and feel? Business; Information Systems Engineering, 2025.

[Co25] Colombo Tosatto, S. et al.: Managing and Anticipating Out-of-Order Events in Online Compliance Monitoring. In: Advanced Information Systems Engineering. Springer Nature Switzerland, pp. 151–166, 2025.

[IK25] Imenkamp, C.; Koschmider, A.: Streaming Process Mining over Realistic Event Streams. In: EMISA 2025. Gesellschaft für Informatik e.V., Bonn, pp. 67–72, 2025.

[Wend] Weber, I. et al.: CCaaS: Online Conformance Checking as a Service, n.d.