Cortado: A dedicated process mining tool for interactive process discovery

Daniel Schuster *, Sebastiaan J. van Zelst, Wil M.P. van der Aalst
Fraunhofer Institute for Applied Information Technology FIT, Sankt Augustin, Germany
RWTH Aachen University, Aachen, Germany

A B S T R A C T

Process discovery is an essential discipline within process mining, which deals with the data-driven generation of insights into operational processes. From event data that capture historical process executions, process discovery algorithms learn a process model describing the execution of the various activities involved. Such discovered models are crucial artifacts used by many process mining techniques. Most existing process discovery approaches can be classified as conventional—they function like a black-box approach and often learn models of poor quality from event data. Cortado is a software tool dedicated to interactive process discovery that lets users gradually learn process models from event data. Cortado leverages domain knowledge and insights extracted from data to develop process models in an interactive manner gradually. We describe Cortado's architecture and functionalities that contribute to the overall goal of interactive process discovery.

Code metadata

Current code version	v1.9.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00057
Legal Code License	GPL 3.0
Code versioning system used	git
Software code languages, tools, and services used	Backend: Python 3.10.x, Frontend: Node.js (successfully tested using Node 16.14.2)
Compilation requirements, operating environments & dependencies	`src/backend/requirements.txt` lists required Python packages of Cortado's backend and `src/frontend/package.json` lists required JavaScript packages of Cortado's frontend
If available Link to developer documentation/manual	https://github.com/cortado-tool/cortado/blob/main/README.md
Support email for questions	daniel.schuster@fit.fraunhofer.de

Software metadata

Current software version	v1.9.0
Permanent link to executables of this version	https://github.com/cortado-tool/cortado/releases/tag/v1.9.0
Legal Software License	GPL 3.0
Computing platforms/Operating Systems	Microsoft Windows 10/11, Linux (successfully tested on Ubuntu 22.04.1 LTS), macOS (Apple Silicon only, successfully tested on macOS 13.1)
Installation requirements & dependencies	Builds are self-contained, i.e., they have no dependencies. Admin rights may be required to run Cortado.
If available, link to user manual - if formally published include a reference to the publication in the reference list	Help pages for selected functionalities are available directly in the tool
Support email for questions	daniel.schuster@fit.fraunhofer.de

* Corresponding author at: Fraunhofer Institute for Applied Information Technology FIT, Sankt Augustin, Germany.
E-mail addresses: daniel.schuster@fit.fraunhofer.de (Daniel Schuster), sebastiaan.van.zelst@fit.fraunhofer.de (Sebastiaan J. van Zelst), wvdaalst@pads.rwth-aachen.de (Wil M.P. van der Aalst).

1. Motivation and significance

The execution of processes in organizations, ranging from production to administrative processes, is often recorded in great detail by information systems. The generated data, referred to as *event data*, contain valuable information about the various

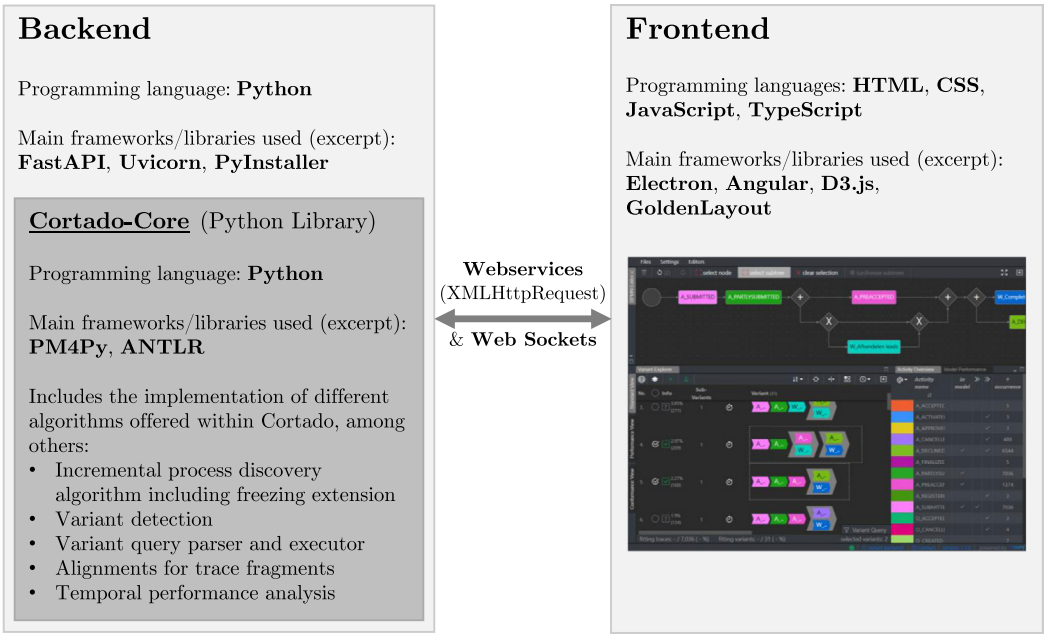


Fig. 1. Overview of the architecture of Cortado, which consists of a backend written in Python and a frontend based on web technologies.

executed process activities, such as temporal and resource information and in which context they have been performed. *Process mining* [1] analyzes such event data to generate insights, such as discovered process models [2], temporal performance diagnostics [3], and conformance checking statistics [4]. Process mining techniques are applied in broad industries, for instance, in healthcare [5,6], manufacturing [7], education [8], auditing [9], and supply chain management [10]. What all these process mining applications ultimately have in common is the goal of supporting process improvements in a data-driven manner.

Process discovery [11] is an important sub-discipline of process mining that deals with learning a process model from event data. Many discovery algorithms exist; we refer to [2,11,12] for overviews. Most algorithms are fully automated—users cannot influence them during their execution, i.e., during the process discovery phase, apart from parameter settings. To this end, interactive process discovery approaches emerged to exploit user knowledge in addition to event data. In [13], an overview of domain knowledge utilizing process discovery is presented.

This paper presents the process mining tool *Cortado*, allowing users, for example, researchers and process analysts, to discover a process model from event data in an incremental, interactive manner. Users can interact with the process discovery algorithm implemented in Cortado at any time, for instance, by selecting process behavior to be added next to the model under construction or by manually editing the model. Cortado combines various process mining algorithms that serve the overall purpose of interactive process discovery. Cortado was first introduced in [14], which covers an early version of Cortado. This paper provides an overarching view of Cortado and its various functionalities added over time. Further, we elaborate on Cortado's impact on research.

We describe Cortado's architecture and functionalities in Section 2, while Section 3 provides illustrative examples of Cortado. Section 4 discusses Cortado's scientific impact. Finally, Section 5 concludes this paper.

2. Software description

Cortado is an end-user process mining tool offered as a standalone desktop application supporting all major operating systems.[1] The source code of Cortado is available on GitHub.[2] We present Cortado's architecture in Section 2.1, followed by an overview of Cortado's functionalities in Section 2.2. We deliberately keep the overview of Cortado's functionalities at a high level of abstraction and refer to specific publications presenting the implemented methods and algorithms in detail.

2.1. Software architecture

Cortado is composed of a Python-based[3] backend and a frontend based on web development technologies. Fig. 1 illustrates the main architecture. The backend and frontend communicate via webservices and web sockets.

2.1.1. Backend

The Python library *Cortado-core* is central to Cortado's backend, cf. Fig. 1. This library implements algorithms and methods for incremental process discovery [15,16], temporal performance analysis [17], process execution variant detection [18], variant querying [19] and conformance checking [20]. Since the Cortado-core library implements essential algorithms for incremental process discovery, it can also be embedded into other software projects and considered an *independent contribution*. Cortado-core uses various other Python libraries, such as PM4Py[4] providing different process mining functionalities and ANTLR [21] for the variant querying functionality.

The backend around Cortado-core is essentially an application programming interface to the frontend, i.e., it bundles functionality from Cortado-core into webservices tailored to the frontend. For this purpose, we use the framework FastAPI.[5] To bundle

1 Available at https://cortado.fit.fraunhofer.de/.
2 https://github.com/cortado-tool/cortado
3 https://www.python.org/
4 https://pm4py.fit.fraunhofer.de/
5 https://github.com/tiangolo/fastapi

Table 1
Excerpt of a real-life event log from a loan application process [22]; each row describes an event containing information about the execution of an activity. Events having the same case ID constitute a process execution. Note that events typically include much more information than shown here.

Case ID	Activity	Timestamp	...
173688	A_SUBMITTED	2011-09-30 22:38:44.546000+00:00	...
173688	A_PARTLYSUBMITTED	2011-09-30 22:38:44.880000+00:00	...
173688	A_PREACCEPTED	2011-09-30 22:39:37.906000+00:00	...
...	...	...	...
205101	A_SUBMITTED	2012-01-30 18:41:02.165000+00:00	...
205101	A_PARTLYSUBMITTED	2012-01-30 18:41:02.300000+00:00	...
205101	A_DECLINED	2012-01-30 18:41:35.174000+00:00	...
...	...	...	...

the backend, including Cortado-core, into a single executable package, we use PyInstaller.[6]

2.1.2. Frontend

The frontend is built as a web application. Thus, we use web development languages like HTML, CSS, JavaScript, and TypeScript. As an application framework we use Angular.[7] For the various visualizations within Cortado, for example, the process model editor or the variant explorer, we use the JavaScript library d3.js.[8] Further, we use the Electron[9] framework to create cross-platform desktop applications.

2.2. Software functionalities

This section provides an overview of Cortado's functionalities. These are divided into: *event data handling*, *incremental process discovery*, *conformance checking* and *temporal performance analysis* (cf. Fig. 2).

2.2.1. Event data handling

Process discovery starts from event data capturing historical process executions; Table 1 provides an example even log. We refer to [23] for an introduction to event data. As real-life event logs often have data quality issues [24,25], event data handling, comprising preprocessing and filtering, is vital when applying process mining [26]. Various techniques exist to cope event log quality issues, for example, [27–29]. Further, real-life event logs often contain a large number of different process executions, making it challenging for process analysts to explore the data as provided. Cortado, therefore, summarizes the event log by grouping individual process executions into *process execution variants* that are visualized within the variant explorer allowing users to visually explore the event log, cf. the variant explorer depicted in Fig. 2. Said variants group individual process executions with identical ordering relationships between the executed process activities. Cortado implements a novel variant definition [18] that extends traditional, sequence-based variants to include and visually represent temporal overlaps of activities, i.e., partially ordered event data.

Although variants group process executions, the number of variants can still be vast, and their visual exploration can be exhausting. Therefore, Cortado provides a query language designed explicitly for querying variants [19]. The query language allows specifying complex control-flow patterns. Executing queries results in variants that satisfy the specified patterns. The focus on control-flow patterns originates from the overall goal of discovering process models—most process model formalisms focus primarily on control-flow structures of activities. As such, users can utilize the query language to explore and filter event logs.

Furthermore, although event data often contain very accurate timestamps (at the millisecond level), it is only sometimes advantageous to analyze processes at this level of accuracy. Therefore, Cortado allows users to *set the time granularity* of the provided event log. Changing the time granularity influences the ordering of activities within variants.

Finally, Cortado allows users to extract manually and mine *frequent variant fragments* from existing variants; further, users can also freely model variants/variant fragments. Said variants and variant fragments are the essential input for the incremental process discovery approach. In short, event data handling in Cortado includes exploring, filtering, preprocessing, querying, and visualization of event data, cf. Fig. 2.

2.2.2. Incremental process discovery

User-selected variants/variant fragments (referred to as variants for simplicity in the following) are the crucial input for the process discovery phase, cf. Fig. 2. By selecting a subset of variants, the user controls the evolution of the process model being discovered. First, an initial process model is discovered using a conventional process discovery algorithm [30] on selected variants or imported into the tool. When an initial process model is in place, the user selects variants from the variant explorer to be incorporated into the process model. The selected variants and the process model are then plugged into the incremental process discovery approach [15], which updates the process model such that the resulting model represents all selected process behavior. The extended process model then serves as an input in the next incremental iteration. During incremental process discovery, users can optionally *freeze model parts* of the process model that is used as an input of the incremental process discovery approach [16]. Frozen model parts are not altered by the incremental process discovery approach when extending/modifying the process model. Thus, *any future model will always contain the frozen sub-model parts*. Note that the above-described iterative approach can be executed multiple times by the user, who controls the progress of process discovery in each iteration by the variants selected and by possible manual changes to the model, as indicated in Fig. 2.

2.2.3. Conformance checking

To compare the discovered process model, including intermediate process models within the incremental discovery approach, with the event log provided, Cortado features conformance checking [4]. Cortado implements *alignments* [20,31], i.e., a state-of-the-art conformance checking technique that provides diagnostics on the mismatches between the observed (i.e., event data) and modeled process behavior (i.e., the process model). Consequently, the user can compare the process model with the event data at any time during incremental process discovery. These diagnostics help users to make informed decisions about which variants to add further and provide an overview of the current process model to which degree the observed behavior from the event log is covered.

2.2.4. Temporal performance analysis

Cortado features temporal performance analysis functionalities [17]. These techniques make it possible, for example, to identify slow process stages or activities, i.e., bottlenecks within the process. Aside from the general importance of these techniques, they are beneficial in interactive process recognition because temporal performance statistics can help users to decide which process behavior to incorporate into the process model incrementally. For example, one is interested in learning a process model

6 https://pyinstaller.org/
7 https://angular.io/
8 https://github.com/d3/d3
9 https://www.electronjs.org/

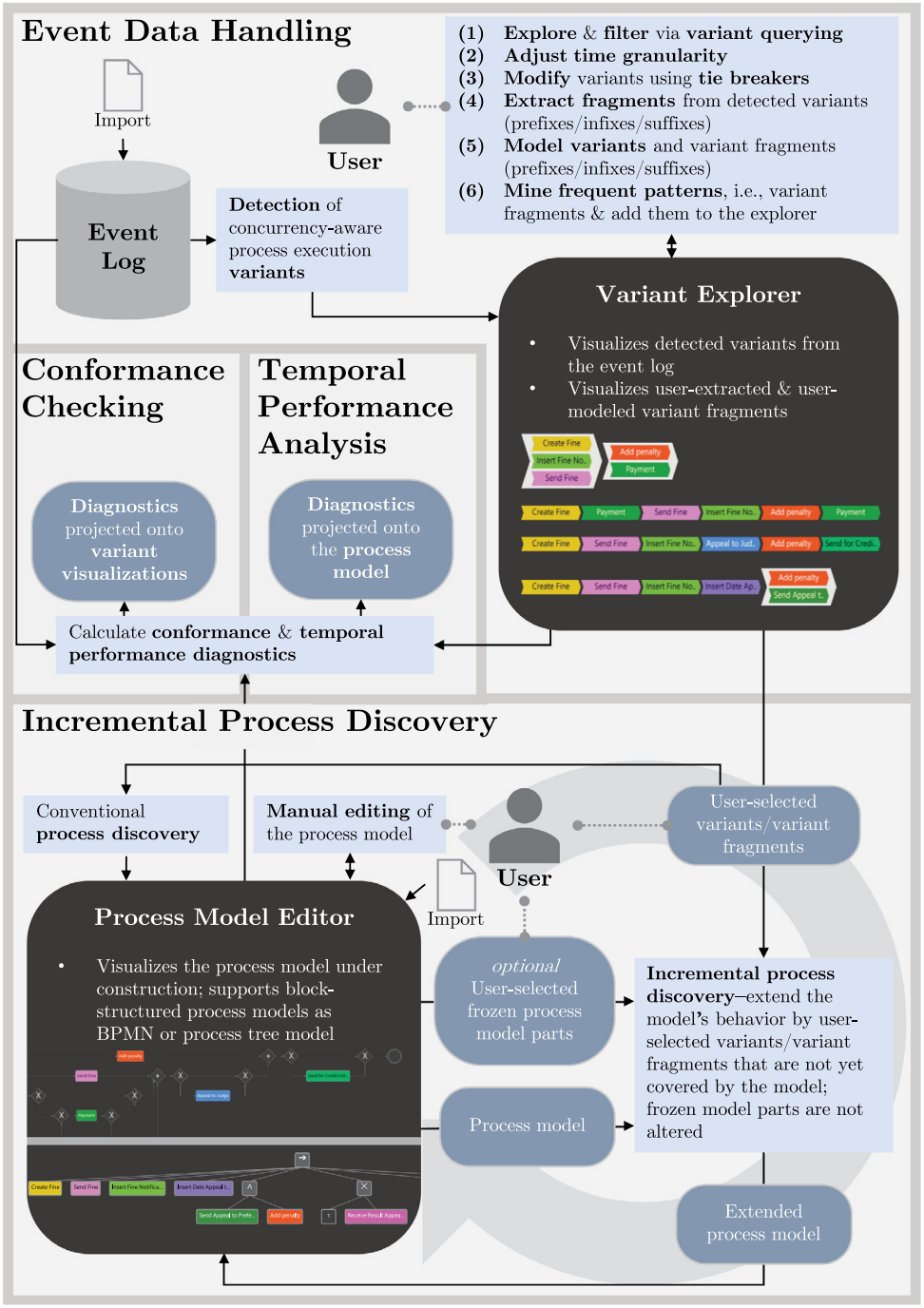
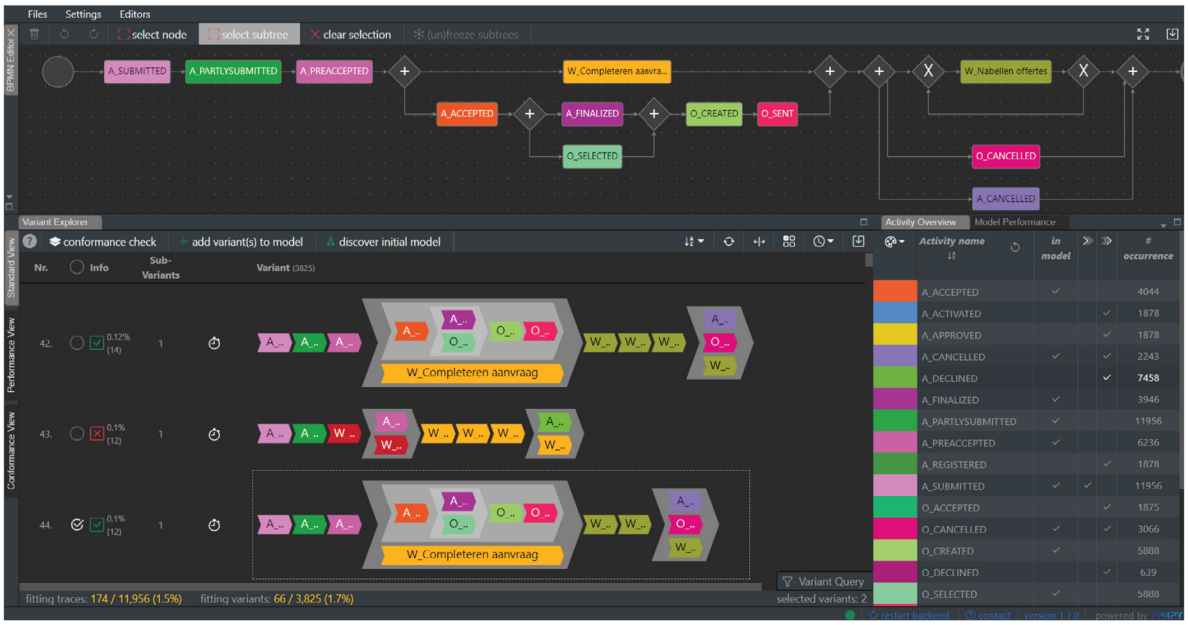


Fig. 2. Simplified overview of Cortado's functionality that can be distinguished in three primary functional themes: (1) event data handling, (2) conformance checking/temporal performance analysis, and (3) incremental process discovery.

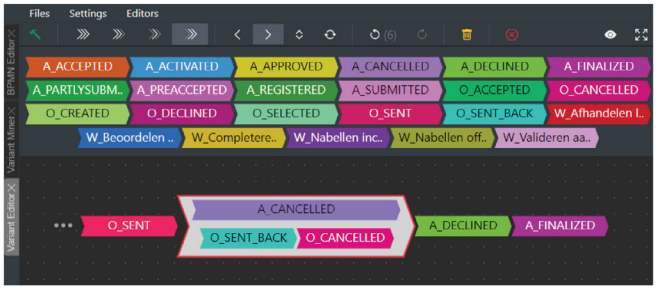
that shows how slow process executions are conducted. Such process models, focusing on a specific subset of process behavior, can be further used for advanced process mining techniques; for example, to simulate particular process changes and their impact on process performance [32].

3. Illustrative example

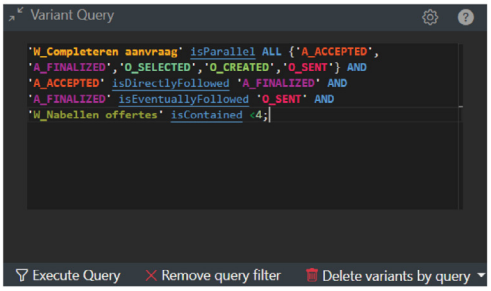
This section introduces Cortado's various functionalities from a user's perspective. Fig. 3 depicts screenshots of its user interface (UI) that show sample functionalities regarding *event data handling* and *incremental process discovery*, cf. Fig. 2. Cortado's main UI is shown in Fig. 3(a). The upper part of the UI contains the process model editor; in the screenshot, an already discovered process model is visualized as a BPMN [33] model. In the lower right part of the UI, an activity summary table is displayed, listing all activities from the event log. The table shows statistics about the frequency, whether the activity is a start or end activity, and whether the process model that the user incrementally discovers contains this activity. To the left of the activity overview, the variant explorer is visible. Initially, only variants detected from the event log are presented. However, users can manually add created variants to this set using the variant editor, depicted in Fig. 3(b). Further, users can mine frequent patterns from the variants by invoking the variant miner, cf. Fig. 3(d). Users can add the mined variant fragments to the list of variants. Fig. 3(c) shows



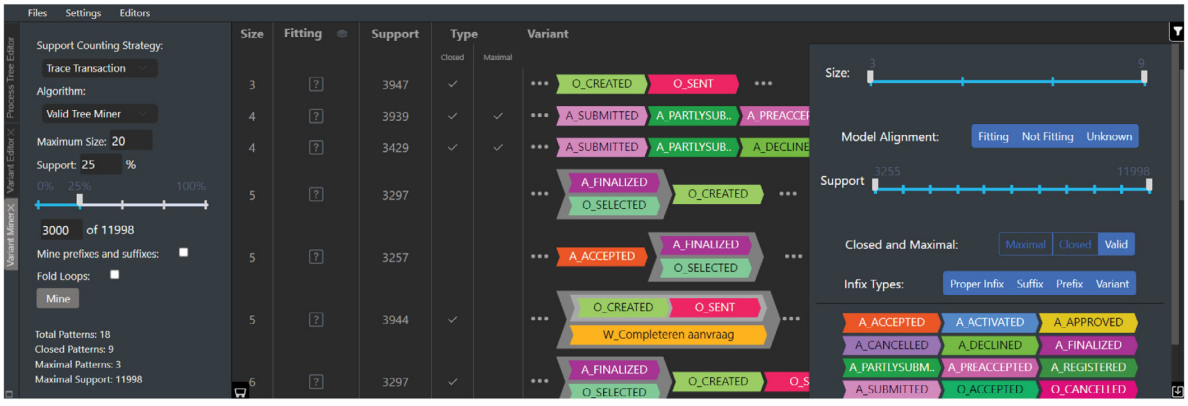
(a) Screenshot of Cortado's overall user interface. The process model editor is located in the upper part, currently showing a BPMN process model. In the lower left, the variant explorer is located, visualizing variants from a real-life event log [22]. The green checkmark respectively the red cross mark indicates whether the variant is represented by the current process model or not. Finally, in the lower right part, an activity overview is presented.



(b) *Variant editor* that allows the manual creation of variants. In the shown screenshot, a variant prefix, i.e., a variant fragment, is modeled, indicated by the three dots in front of the variant.

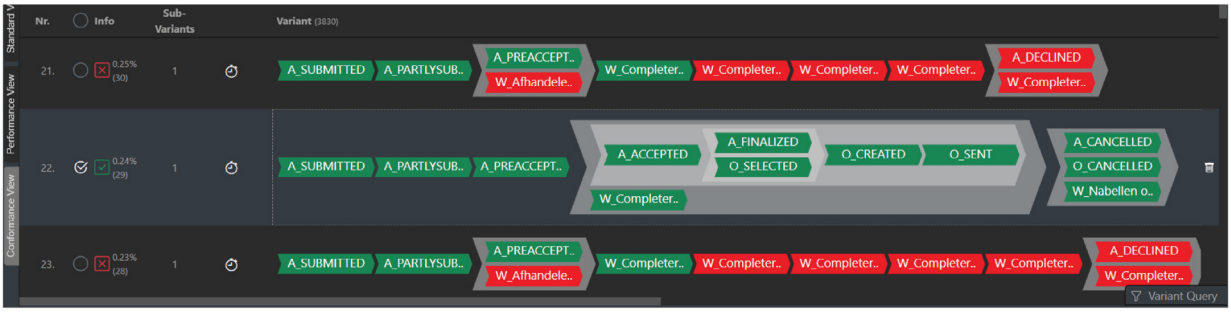


(c) *Variant query* editor allowing to specify control-flow patterns. Executing a query results in an updated list of variants in the variant explorer satisfying the constraints.

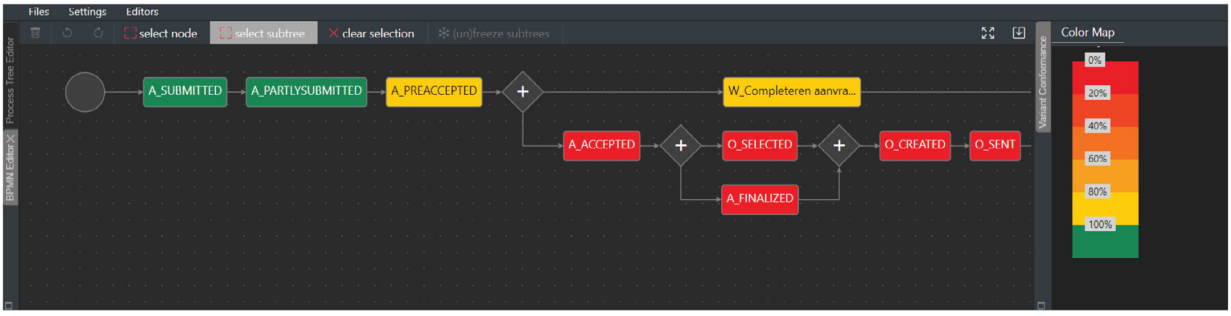


(d) *Variant miner* that allows for frequent pattern mining from variants, i.e., prefix/infix/postfix patterns. Frequent patterns are shown in the middle, similar to the variant explorer. On the left, users can configure settings for frequent pattern mining. On the right, filter options provide options to explore the result set.

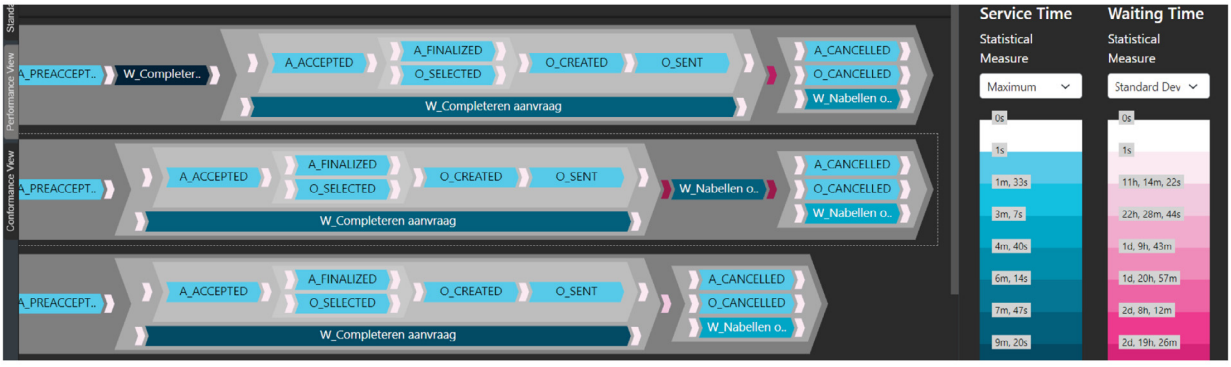
Fig. 3. Screenshots from Cortado showing selected functionalities from *event data handling* and *incremental process discovery*, cf. Fig. 2.



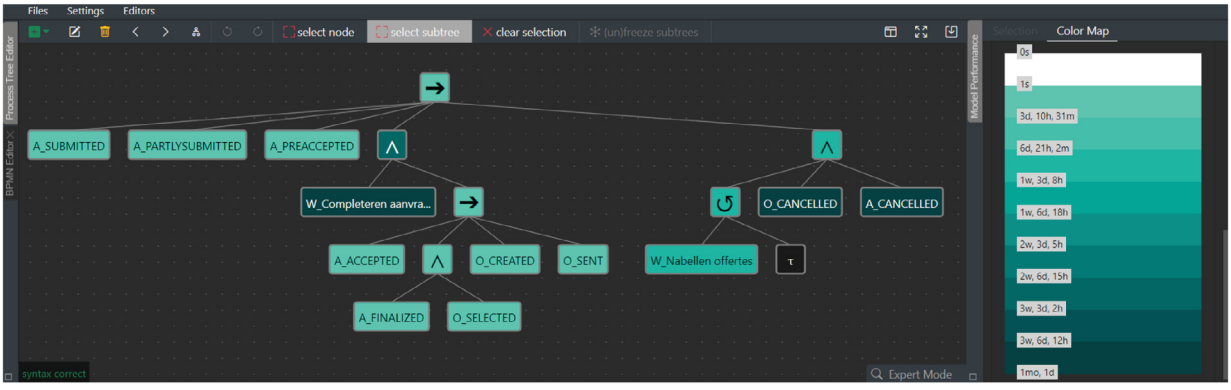
(a) *Conformance view* of the variant explorer—projecting conformance diagnostics onto variants; red highlighted activities indicate that these could not have been replayed in the process model



(b) *Conformance view* of the process model editor—projection of conformance diagnostics on models; activities not highlighted in green indicate that they were not performed for specific process executions recorded in the event log



(c) *Performance view* of the variant explorer—projecting temporal performance statistics, i.e., service and waiting time, onto variants in the variant explorer



(d) *Performance view* of the process model editor—projection of temporal performance statistics, in this case, waiting times, onto process models

Fig. 4. Screenshots from Cortado showing selected functionalities from *conformance checking* and *temporal performance analysis*.

an example of the query language implemented in Cortado. The query language facilitates the exploration, filtering, and finding of variants. To incrementally extend a process model, users select variants in the variant explorer by the checkbox icon and start one iteration of the incremental process discovery approach via a button 'add variant(s) to model', cf. Fig. 3(a).

Fig. 4 depicts screenshots showing functionalities from *conformance checking* and *temporal performance analysis*. In general, said techniques use the event log and a (discovered) process model as input. Conformance checking techniques [4] compare modeled process behavior, i.e., a process model, with recorded process behavior, i.e., event data, to find deviations between the two. Cortado visualizes the resulting conformance checking diagnostics by projecting them onto the variant visualizations, cf. Fig. 4(a), but also on the process model. Fig. 4(b) depicts the variant explorer in conformance view; colors indicate deviations between the process model and the observed process execution variants, i.e., behavior that should not have happened at this point according to the process model. Finally, Fig. 4(b) shows a process model where the color grading indicates deviations, i.e., the redder the color, the more often this activity in the model has been skipped in the recorded process behavior, i.e., the event data.

4. Impact

This section briefly reviews the impact of Cortado on interactive process discovery and process mining in general. We outline existing publications of algorithms and methods implemented in Cortado and highlight emerging research questions arising from Cortado.

Cortado serves as a platform for interactive process discovery techniques and related research contributions. Various novel algorithms and approaches have been implemented within Cortado [15–20]. Especially for interactive process discovery techniques, it is of great importance to have end-user-focused implementations to evaluate these techniques with users. Furthermore, interactive process mining must be considered as a more comprehensive approach that is only complete and viable when different techniques are combined, as indicated in Section 2.2. In this way, Cortado contributes to considering and implementing interactive process discovery as a comprehensive and holistic approach. Since Cortado is a mature prototype, real-world use cases can benefit from the tool and use it beneficially. Therefore, there is a broad possibility to use Cortado outside of research.

By tightly coupling different methods and algorithms with the overarching goal of interactive process discovery in Cortado, interesting new research questions are emerging. For instance, insights into the use of interactive process discovery approaches are still little researched [34,35]. To this end, Cortado is currently serving as an interactive process discovery tool in an ongoing use case study; the study aims to identify the benefits of interactive process discovery in a concrete use case.

While process mining generally encompasses many algorithms and techniques, process analysts' application of these algorithms and techniques in practice and their needs and requirements for process mining techniques have been little studied. Recent work started to explore how process analysts work by identifying practices [36] and challenges [37,38]. To this end, Cortado offers unique opportunities for studies to explore how users, i.e., process analysts, can benefit from interactive process discovery techniques in process mining projects.

Although Cortado focuses on interactive process discovery, it offers users novel means to explore, analyze and filter event data. By providing variant visualizations that take concurrency into account, and a corresponding query language tailored to these variants, new opportunities arise for users to analyze event data. In addition, supporting the `xes` file format [39], a standard for event logs, ensures operability with other process mining tools. For instance, users can use Cortado as an exploration and filtering-only tool, exporting the preprocessed data and loading it into other process mining tools for further analysis. The same applies to the functionalities of conformance checking/temporal performance analysis. If users already have a process model, they can import it into Cortado and make use of the implemented techniques.

Furthermore, as depicted in Fig. 1, Cortado's backend builds around the Python library Cortado-core that implements all relevant algorithms [15–20]. Thus, Cortado-core can be considered an independent contribution that can be embedded as a library within other process mining software projects and be used for research projects, for instance, to compare to other discovery algorithms.

5. Conclusion

Cortado combines various techniques and algorithms into a dedicated process mining tool for interactive process discovery. We introduced Cortado's three main functional areas: event data handling, conformance checking/ temporal performance analysis, and incremental process discovery, and have discussed how these functionalities benefit the overall goal of interactive process discovery. Since Cortado is designed for end-users, primarily process analysts and researchers in process mining, we presented selected screenshots showing an excerpt of the functionalities from the user's point of view. Further, we introduced its architecture, divided into a frontend and a backend that includes a Python library called Cortado-core; this library implements all relevant process mining algorithms and approaches.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgments

We thank everyone who contributed to Cortado's development (sorted alphabetically): Lars Dietrich, Niklas Föcking, Gero J. Kolhof, Michael Martini, Minh Nghia Phan, and Lukas Schade.

References

[1] van der Aalst WMP. Process mining: data science in action. Springer; 2016, http://dx.doi.org/10.1007/978-3-662-49851-4.
[2] Augusto A, Conforti R, Dumas M, La Rosa M, Maggi FM, Marrella A, Mecella M, Soo A. Automated discovery of process models from event logs: Review and benchmark. IEEE Trans Knowl Data Eng 2019;31(4):686–705. http://dx.doi.org/10.1109/TKDE.2018.2841877.
[3] Park G, Song M. Predicting performances in business processes using deep neural networks. Decis Support Syst 2020;129:113191. http://dx.doi.org/10.1016/j.dss.2019.113191.
[4] Carmona J, van Dongen B, Solti A, Weidlich M. Conformance checking. Springer; 2018, http://dx.doi.org/10.1007/978-3-319-99414-7.
[5] Rojas E, Munoz-Gama J, Sepúlveda M, Capurro D. Process mining in healthcare: A literature review. J Biomed Inform 2016;61:224–36. http://dx.doi.org/10.1016/j.jbi.2016.04.007.
[6] Mans RS, van der Aalst WMP, Vanwersch RJB. Process mining in healthcare. Cham: Springer; 2015, http://dx.doi.org/10.1007/978-3-319-16071-9.

[7] Yang H, Park M, Cho M, Song M, Kim S. A system architecture for manufacturing process analysis based on big data and process mining techniques. In: 2014 IEEE international conference on big data (big data). IEEE; 2014, p. 1024–9. http://dx.doi.org/10.1109/BigData.2014.7004336.
[8] Bogarín A, Cerezo R, Romero C. A survey on educational process mining. WIREs Data Min Knowl Discov 2018;8(1). http://dx.doi.org/10.1002/widm.1230.
[9] Jans M, Alles M, Vasarhelyi M. The case for process mining in auditing: Sources of value added and areas of application. Int J Account Inform Syst 2013;14(1):1–20. http://dx.doi.org/10.1016/j.accinf.2012.06.015.
[10] Lau H, Ho G, Zhao Y, Chung N. Development of a process mining system for supporting knowledge discovery in a supply chain network. Int J Prod Econ 2009;122(1):176–87. http://dx.doi.org/10.1016/j.ijpe.2009.05.014.
[11] Augusto A, Carmona J, Verbeek E. Advanced process discovery techniques. In: van der Aalst WMP, Carmona J, editors. Process mining handbook. Lecture notes in business information processing, vol. 448, Springer; 2022, p. 76–107. http://dx.doi.org/10.1007/978-3-031-08848-3_3.
[12] van Dongen BF, Alves de Medeiros AK, Wen L. Process mining: Overview and outlook of Petri net discovery algorithms. In: Jensen K, van der Aalst WMP, editors. Transactions on Petri nets and other models of concurrency II. Lecture notes in computer science, vol. 5460, Springer; 2009, p. 225–42. http://dx.doi.org/10.1007/978-3-642-00899-3_13.
[13] Schuster D, van Zelst SJ, van der Aalst WMP. Utilizing domain knowledge in data-driven process discovery: A literature review. Comput Ind 2022;137. http://dx.doi.org/10.1016/j.compind.2022.103612.
[14] Schuster D, van Zelst SJ, van der Aalst WMP. Cortado—An interactive tool for data-driven process discovery and modeling. In: Buchs D, Carmona J, editors. Application and theory of Petri nets and concurrency. Lecture notes in computer science, vol. 12734, Springer; 2021, p. 465–75. http://dx.doi.org/10.1007/978-3-030-76983-3_23.
[15] Schuster D, van Zelst SJ, van der Aalst WMP. Incremental discovery of hierarchical process models. In: Dalpiaz F, Zdravkovic J, Loucopoulos P, editors. Research challenges in information science. Lecture notes in business information processing, vol. 385, Cham: Springer; 2020, p. 417–33. http://dx.doi.org/10.1007/978-3-030-50316-1_25.
[16] Schuster D, van Zelst SJ, van der Aalst WMP. Freezing sub-models during incremental process discovery. In: Ghose A, Horkoff J, Silva Souza VE, Parsons J, Evermann J, editors. Conceptual modeling. Lecture notes in computer science, vol. 13011, Springer; 2021, p. 14–24. http://dx.doi.org/10.1007/978-3-030-89022-3_2.
[17] Schuster D, Schade L, van Zelst SJ, van der Aalst WMP. Temporal performance analysis for block-structured process models in Cortado. In: de Weerdt J, Polyvyanyy A, editors. Intelligent information systems. Lecture notes in business information processing, vol. 452, Springer; 2022, p. 110–9. http://dx.doi.org/10.1007/978-3-031-07481-3_13.
[18] Schuster D, Schade L, van Zelst SJ, van der Aalst WMP. Visualizing trace variants from partially ordered event data. In: Munoz-Gama J, Lu X, editors. Process Mining Workshops. Lecture notes in business information processing, vol. 433, Springer; 2022, p. 34–46. http://dx.doi.org/10.1007/978-3-030-98581-3_3.
[19] Schuster D, Martini M, van Zelst SJ, van der Aalst WMP. Control-flow-based querying of process executions from partially ordered event data. In: Troya J, Medjahed B, Piattini M, Yao L, Fernández P, Ruiz-Cortés A, editors. Service-oriented computing. Lecture notes in computer science, vol. 13740, Cham: Springer; 2022, p. 19–35. http://dx.doi.org/10.1007/978-3-031-20984-0_2.
[20] Schuster D, Föcking N, van Zelst SJ, van der Aalst WMP. Conformance checking for trace fragments using infix and postfix alignments. In: Sellami M, Ceravolo P, Reijers HA, Gaaloul W, Panetto H, editors. Cooperative information systems. Lecture notes in computer science, vol. 13591, Springer; 2022, p. 299–310. http://dx.doi.org/10.1007/978-3-031-17834-4_18.
[21] Parr TJ, Quong RW. ANTLR: A predicated-LL(k) parser generator. Softw - Pract Exp 1995;25(7):789–810. http://dx.doi.org/10.1002/spe.4380250705.
[22] van Dongen BF. BPI Challenge 2012 - event log, Eindhoven University of Technology, https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.
[23] de Weerdt J, Wynn MT. Foundations of process event data. In: van der Aalst WMP, Carmona J, editors. Process Mining Handbook. Lecture notes in business information processing, vol. 448, Springer; 2022, p. 193–211. http://dx.doi.org/10.1007/978-3-031-08848-3_6.
[24] Martin N. Data quality in process mining. In: Fernandez-Llatas C, editor. Interactive process mining in healthcare. Health informatics, Springer; 2021, p. 53–79. http://dx.doi.org/10.1007/978-3-030-53993-1_5.
[25] Bose RJC, Mans RS, van der Aalst WMP. Wanna improve process mining results? In: 2013 IEEE symposium on computational intelligence and data mining. IEEE; 2013, p. 127–34. http://dx.doi.org/10.1109/CIDM.2013.6597227.
[26] van Eck ML, Lu X, Leemans SJJ, van der Aalst WMP. PM^2: A process mining project methodology. In: Zdravkovic J, Kirikova M, Johannesson P, editors. Advanced information systems engineering. Lecture notes in computer science, vol. 9097, Springer; 2015, p. 297–313. http://dx.doi.org/10.1007/978-3-319-19069-3_19.
[27] Suriadi S, Andrews R, ter Hofstede A, Wynn MT. Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. Inf Syst 2017;64:132–50. http://dx.doi.org/10.1016/j.is.2016.07.011.
[28] Andrews R, Suriadi S, Ouyang C, Poppe E. Towards event log querying for data quality. In: Panetto H, Debruyne C, Proper HA, Ardagna CA, Roman D, Meersman R, editors. On the move to meaningful internet systems. OTM 2018 conferences. Lecture notes in computer science, vol. 11229, Springer; 2018, p. 116–34. http://dx.doi.org/10.1007/978-3-030-02610-3_7.
[29] Tax N, Sidorova N, van der Aalst WMP. Discovering more precise process models from event logs by filtering out chaotic activities. J Intell Inf Syst 2019;52(1):107–39. http://dx.doi.org/10.1007/s10844-018-0507-6.
[30] Leemans SJJ, Fahland D, van der Aalst WMP. Discovering block-structured process models from event logs - a constructive approach. In: Application and theory of Petri Nets and concurrency, vol. 7927. p. 311–29. http://dx.doi.org/10.1007/978-3-642-38697-8_17.
[31] Adriansyah A. Aligning observed and modeled behavior (Ph.D. thesis), Technische Universiteit Eindhoven; 2014, http://dx.doi.org/10.6100/IR770080.
[32] Măruşter L, van Beest NRTP. Redesigning business processes: a methodology based on simulation and process mining techniques. Knowl Inf Syst 2009;21(3):267–97. http://dx.doi.org/10.1007/s10115-009-0224-0.
[33] Chinosi M, Trombetta A. BPMN: An introduction to the standard. Comput Stand Interf 2012;34(1):124–34. http://dx.doi.org/10.1016/j.csi.2011.06.002.
[34] Benevento E, Aloini D, van der Aalst WMP. How can interactive process discovery address data quality issues in real business settings? Evidence from a case study in healthcare. J Biomed Inform 2022;130:104083. http://dx.doi.org/10.1016/j.jbi.2022.104083.
[35] Benevento E, Dixit PM, Sani MF, Aloini D, van der Aalst WMP. Evaluating the effectiveness of interactive process discovery in healthcare: A case study. In: Di Francescomarino C, Dijkman R, Zdun U, editors. Business process management workshops. Lecture notes in business information processing, vol. 362, Springer; 2019, p. 508–19. http://dx.doi.org/10.1007/978-3-030-37453-2_41.
[36] Zerbato F, Soffer P, Weber B. Initial insights into exploratory process mining practices. In: Polyvyanyy A, Wynn MT, van Looy A, Reichert M, editors. Business process management forum. Lecture notes in business information processing, vol. 427, Springer; 2021, p. 145–61. http://dx.doi.org/10.1007/978-3-030-85440-9_9.
[37] Zimmermann L, Zerbato F, Weber B. Process mining challenges perceived by analysts: An interview study. In: Augusto A, Gill A, Bork D, Nurcan S, Reinhartz-Berger I, Schmidt R, editors. Enterprise, business-process and information systems modeling. Lecture notes in business information processing, vol. 450, Springer; 2022, p. 3–17. http://dx.doi.org/10.1007/978-3-031-07475-2_1.
[38] Martin N, Fischer DA, Kerpedzhiev GD, Goel K, Leemans SJJ, Röglinger M, van der Aalst WMP, Dumas M, La Rosa M, Wynn MT. Opportunities and challenges for process mining in organizations: Results of a delphi study. Bus Inform Syst Eng 2021;63(5):511–27. http://dx.doi.org/10.1007/s12599-021-00720-0.
[39] Acampora G, Vitiello A, Di Stefano B, van der Aalst WMP, Gunther C, Verbeek E. IEEE 1849: The XES standard: The second IEEE standard sponsored by IEEE computational intelligence society [society briefs]. IEEE Comput Intell Mag 2017;12(2):4–8. http://dx.doi.org/10.1109/MCI.2017.2670420.