A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light greenish-blue. They are positioned diagonally, with the blue one partially covering the green one.

Enhancing the Process
Mining Visualization
Tool by adding the
Inductive Miner and
improving the UI



Agenda

- Motivation
- Grundlagen
- Architektur
- Software Development Prozess
- Demo
- Evaluation
- Herausforderungen
- Fazit



Motivation

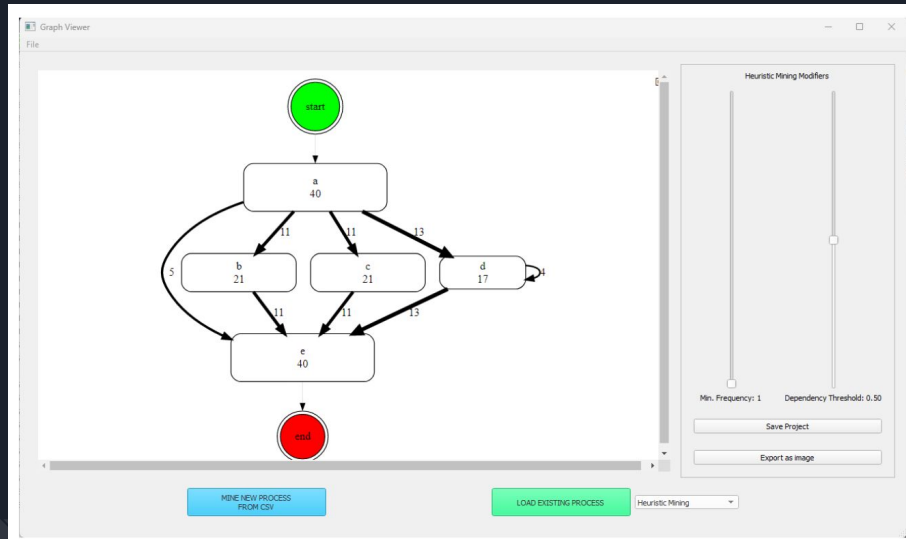
- Wichtigkeit Process Mining [1,2]
- Die meisten Tools ohne Lizenz nicht nutzbar
 - Celonis [13]
 - Disco [10]
 - PMTK [11]
 - Ausnahme ProM 6 [12]
- Verbesserung des Process Mining Visualization Tools [3,4]
 - Erweiterung der analytischen Fähigkeiten
 - Modernisierung des UI



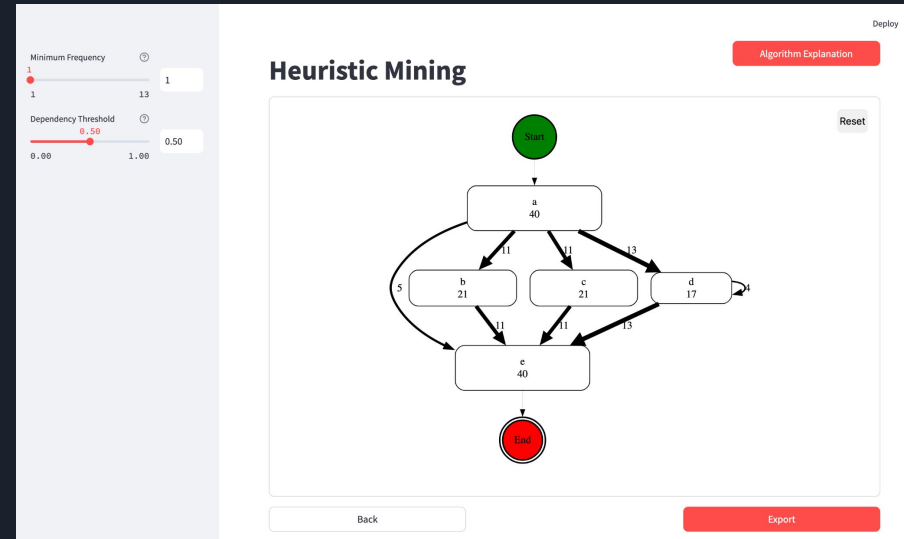
Grundlagen - Verbesserung UI

- Nutzung einer aktuellen Library
- streamlit [8]
 - Einfach zu Verwenden
 - Keine Frontend Kenntnisse erforderlich
- Modernes und responsive User Interface


Grundlagen - Verbesserung UI



Process Mining Visualization Tool 0.1.0 [3]



Process Mining Visualization Tool 0.2.0



Grundlagen - Fehlerbehebungen und Performance Verbesserung

- Fehlerbehebung
 - Alle Graphen haben einen Start- und Endpunkt
 - Drücken von Nodes funktioniert im Bereich der Nodes
 - Fuzzy Miner - Edge filtering
 - Heuristic Miner - Bereich der Minimumfrequenz
 - Delimiter Detection
 - etc.
- Performance Verbesserungen
 - Keine mehrfache Verarbeitung von gleichen Traces
 - Nutzung von Numpy Operationen
 - Verarbeitung von größeren Datensätzen

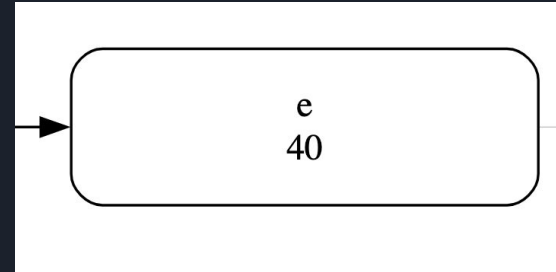
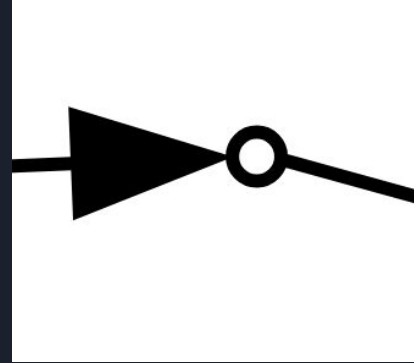


Grundlagen - Inductive Miner [1,5]

- rekursiver Algorithmus ohne Berücksichtigung von Frequenz
- Log Filtering für bessere Modelle
- sucht Partitionen in directly-follows Graphen
- erstellt sound Process Models
- entdeckt mehr Beziehung zwischen Events
 - exklusive Entscheidung
 - Sequenz
 - parallele Ausführung
 - Loop Ausführung
- Ablauf:
 - überprüfen von BaseCases
 - suchen nach Partitionen
 - Cut Detection
 - Log Splitting
 - rekursiver Aufruf mit Sublogs
 - Fall Through

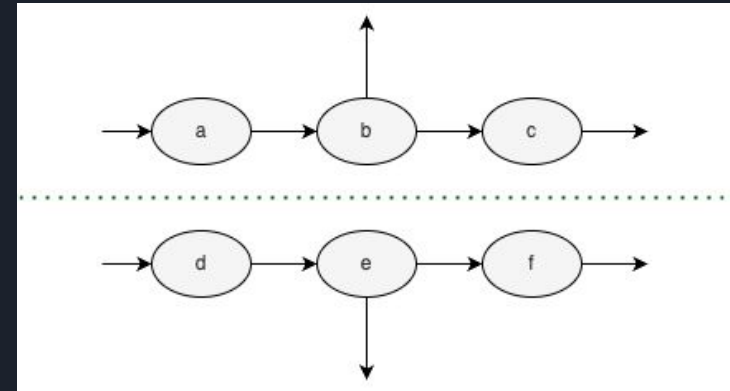
Grundlagen - Base Cases [1,5]

- 2 Base Cases
- Leerer Log
 - keine Events werden ausgeführt
 - silent transition
- Ein Event Trace
 - ein Event wird ausgeführt
 - wird mit Event Namen dargestellt



Grundlagen - Exclusive Cut [1,5]

- Ausführung nur einer Partition
- ungeordnete Partition
- Bedingungen
 - keine Kante zwischen Partitionen





Grundlagen - Exclusive Split [1,5]

- Traces werden als ganzes zugeordnet
- Vorgehen:
 - Finde Partition vom ersten Event
 - (Optional) Überprüfe restliche Events

Log = [<abc>, <def>, <ab>, <de>]

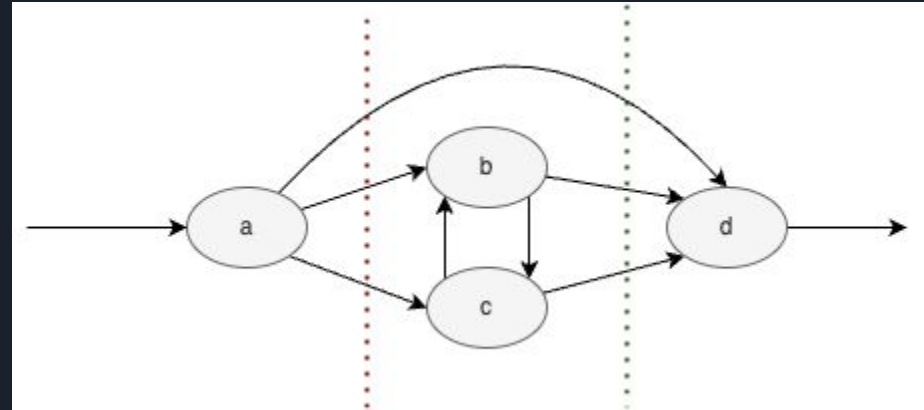
Partitionen = [{a,b,c}, {d,e,f}]

Log_1 = [<abc>, <ab>]

Log_2 = [<def>, <de>]

Grundlagen - Sequence Cut [1,5]

- Sequenz der Ausführung
- geordnete Partition
- Bedingungen
 - Pfad von vorherige in nachfolgende
 - kein Pfad von nachfolgende in vorherige





Grundlagen - Sequence Split [1,5]

- Traces werden als geteilt
- Subtraces werden zugeordnet
- Vorgehen (für jeden Trace):
 - a. Starte bei 1.Partition und Event
 - b. solange Event in Partition
 - zu Subtrace hinzufügen
 - nächstes Event wählen
 - c. Wenn Event nicht in Partition
 - Subtrace zur Partition
 - nächste Partition wählen
 - zurück zu Schritt b
 - d. Ausführung bis Ende vom Trace

$L = [<abcd>, <ad>, <acbd>]$

Partitionen = $\{\{a\}, \{b,c\}, \{d\}\}$

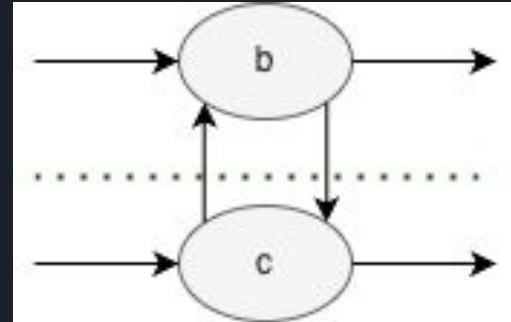
$L_1 = [<a>]$

$L_2 = [<bc>, <>, <cb>]$

$L_3 = [<d>]$

Grundlagen - Parallel Cut [1,5]

- Parallele Ausführung
- ungeordnete Partition
- Bedingungen:
 - Kanten zu allen Knoten außerhalb der Partition
 - Start- und Endevent in Partition





Grundlagen - Parallel Split [1,5]

- Traces werden projiziert
- Für jede Partition eigene Projektion

$L = [<bc>, <cb>, , <c>]$

Partition = $[{\{b\}}, {\{c\}}]$

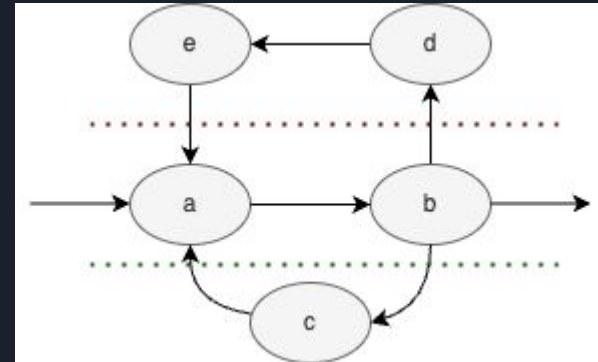
$L_1 = [, <>]$

$L_2 = [<c>, <>]$

- Vorgehen
 - Event nicht in Partition werden gelöscht
 - durchgeführt pro Trace und Partition

Grundlagen - Loop Cut [1,5]

- Unterscheidung do and redo Teil
 - mehrere Redo Teile möglich
 - teilweise geordnete Partition
-
- Bedingungen
 - Start- und Endevents in 1. Partition
 - Redo Teile sind exklusive Partitionen
 - Redo Teile starten bei Endevents
 - Redo Teile enden bei Startevents





Grundlagen - Loop Split [1,5]

- teilt Traces in Subtraces

$L = [<abcab>, <ab>, <abcabcab>, <abdeab>]$


Partitionen = $\{\{a,b\}, \{c\}, \{d,e\}\}$

- Vorgehen:
 - ähnlich Sequenz Split
 - Event nicht in Partition, dann muss Partition gesucht werden

$L_1 = [<ab>]$

$L_2 = [<c>]$

$L_3 = [<de>]$

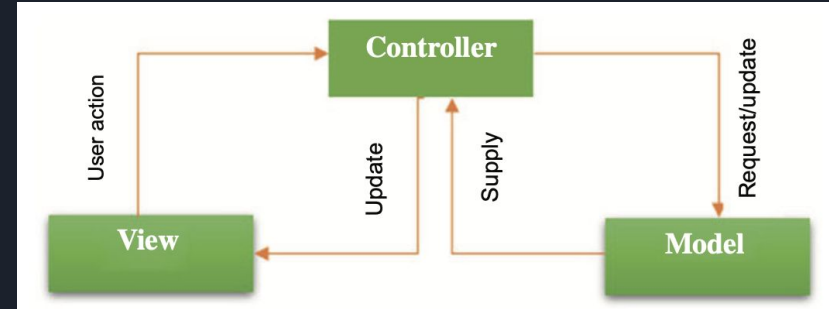


Grundlagen - Fall Through

- Wenn keine Partition gefunden wurde, wird der Fall Through angewendet
- Empty Trace Fall Through
 - Wenn leerer Trace in Log
 - Exklusive Entscheidung mit silent activity
- One Event Fall Through
 - Wenn Log ein Event hat, das mehrmals aufgerufen wird
 - Loop mit Event und silent activity
- Flower Model Fall Through
 - Letztes Mittel
 - Loop mit silent activity und allen Events

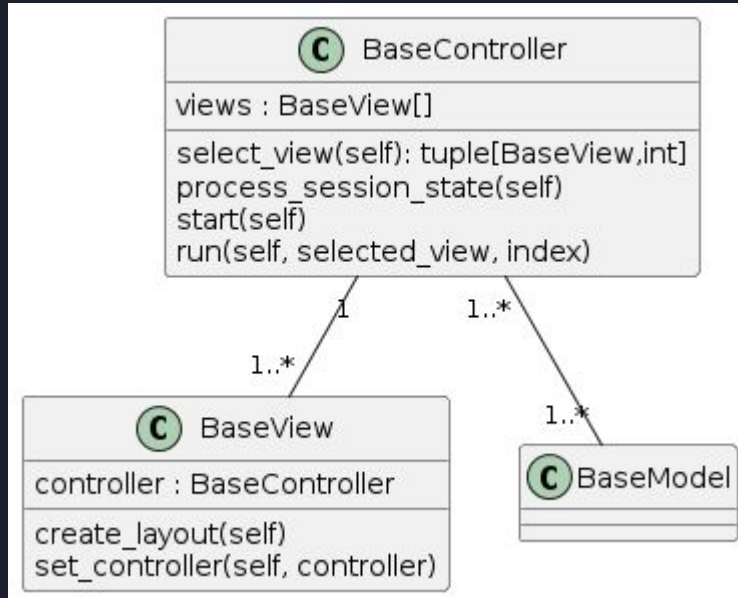
Architektur - MVC Pattern[6,7]

- 3 Komponentnen:
 - Model
 - View
 - Controller
- Vorteile:
 - Testbarkeit
 - Erweiterbarkeit
 - Separation of Concerns

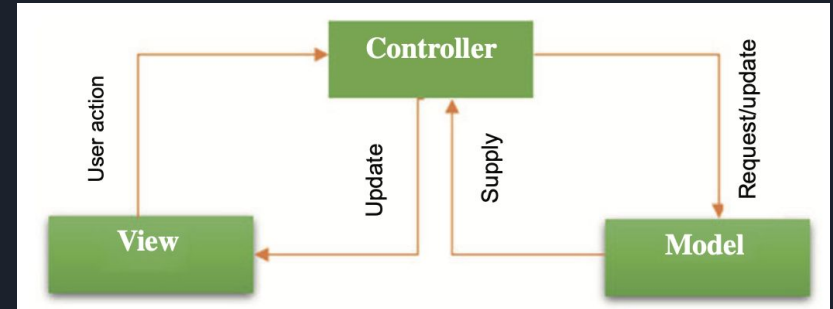


MVC Pattern Diagramm [6]

Architektur - MVC Pattern



MVC Pattern UML Klassen Diagramm



MVC Pattern Diagramm [6]



Software Development Prozess - GitHub

- Speicherung von Code
- Issues
 - Dokumentation von Features
 - Dokumentation von aufkommenden Fehlern
 - Dokumentation von Verbesserungen
- Pull Requests
 - Größere Änderungen in einem eigenen Branch
 - Mergen nach Fertigstellung der Änderung
 - Dokumentation der Änderungen

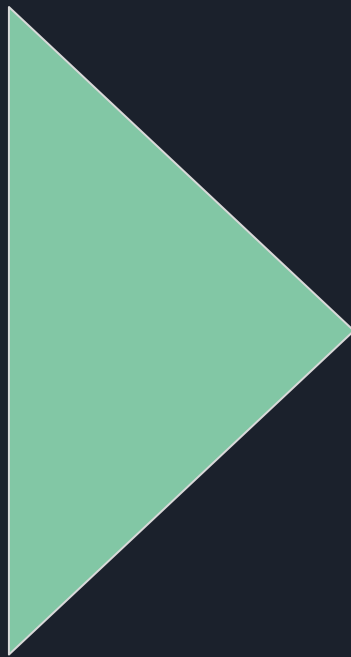


Software Development Prozess - Testing

- Unit testing wurde während der Entwicklung eingesetzt.
- Logik der Applikation wurde automatisiert getestet.
 - Mining Algorithmen
 - Modelle des MVC Patterns
- Views und Controller wurden nicht automatisiert getestet.



Demo





Evaluation - Verbesserungen

- moderneres User Interface
- bessere Performance
- Projektstruktur, Einteilung in Module
- Besser Erweiterbar

	Previous Implementation	Current Implementation
Heuristic View	134	21
Heuristic Controller	28	37
Fuzzy View	187	37
Fuzzy Controller	32	47

Table 6.1: Line of Codes in class files



Herausforderungen

- Grenzen von Streamlit
 - Layout
 - Styling
 - Session State
- Performance der Applikation
 - Caching
 - Runtime Komplexität
- Erstellung des interaktiven Graphen mit React
 - Dokumentation von d3-graphviz [9]
 - Performance beim Layout und rendern
 - Zerstören von alten Instanzen



Fazit

- Streamlit
 - einfache Nutzung
 - Community Support
- verbessertes Open-Source Process Mining Tool wurde erstellt
 - modernere UI
 - verbesserte analytische Fähigkeiten
- Tool kann in Zukunft einfach erweitert werden
 - Viele Schnittstellen für Erweiterung



Literatur

- [1] Leemans, S. J. J., Fahland, D., and van der Aalst, W. M. P. Discovering block-structured process models from event logs - a constructive approach. In *Application and Theory of Petri Nets and Concurrency*, J.-M. Colom and J. Desel, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 311–329.
- [2] van der Aalst, W. M. P. *Process Mining Discovery, Conformance and Enhancement of Business Processes*. Springer Verlag Berlin Heidelberg, 2011.
- [3] Chen, A. *A Python Desktop App for Business. Process Mining and Visualization*. Bachelor thesis, University of Vienna, 2023.
- [4] Krasniqi, A. *Fuzzy Miner as an additional Process Mining Algorithm for a Process Mining Visualization Tool in Python*. Bachelor thesis, University of Vienna, 2024.
- [5] Leemans, S. *Robust process mining with guarantees*. PhD thesis, Technische Universiteit Eindhoven, 2017.
- [6] Rana, M. E., and Saleh, O. S. Chapter 15 - high assurance software architecture and design. In *System Assurances*, P. Johri, A. Anand, J. Vain, J. Singh, and M. Quasim, Eds., *Emerging Methodologies and Applications in Modelling*. Academic Press, 2022, pp. 271–285.
- [7] Grove, R. F., and Ozkan, E. The mvc-web design pattern. In *Proceedings of the 7th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST*, SciTePress, 2011, pp. 127–130



Literatur

[8] Streamlit. Streamlit. <https://streamlit.io/>, 2024. Accessed on 2024-06-26.

[9] Jacobsson, M. d3-graphviz. <https://www.npmjs.com/package/d3-graphviz>, 2024. Accessed on 2024-06-26.

[10] Fluxicon. Process mining and automated process discovery software for professionals - fluxicon disco. <https://fluxicon.com/disco/>, 2024. Accessed on 2024-07-13.

[11] Process Intelligence Solutions. Process intelligence solutions. <https://processintelligence.solutions/>, 2024. Accessed on 2024-07-13.

[12] ProMTools. Prom tools. <https://promtools.org/>, 2024. Accessed on 2024-07-13.

[13] Celonis. Process mining und execution management software | celonis. <https://www.celonis.com/de/>, 2024. Accessed on 2024-07-13.

Vielen Dank für Ihre
Aufmerksamkeit

