Christian-Albrechts-Universität zu Kiel

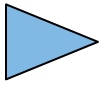
Advanced Process Mining

Prof. Dr. Agnes Koschmider

Lecture 3: Process Mining Projects with Directly-Follows Graph

Lecture Overview

Christian-Albrechts-Universität zu Kiel



0	Organization and Introduction
I	Process Discovery
II	Process Conformance
III	Predictive Process Mining
IV	Event Log Preparation
V	Practical Tasks

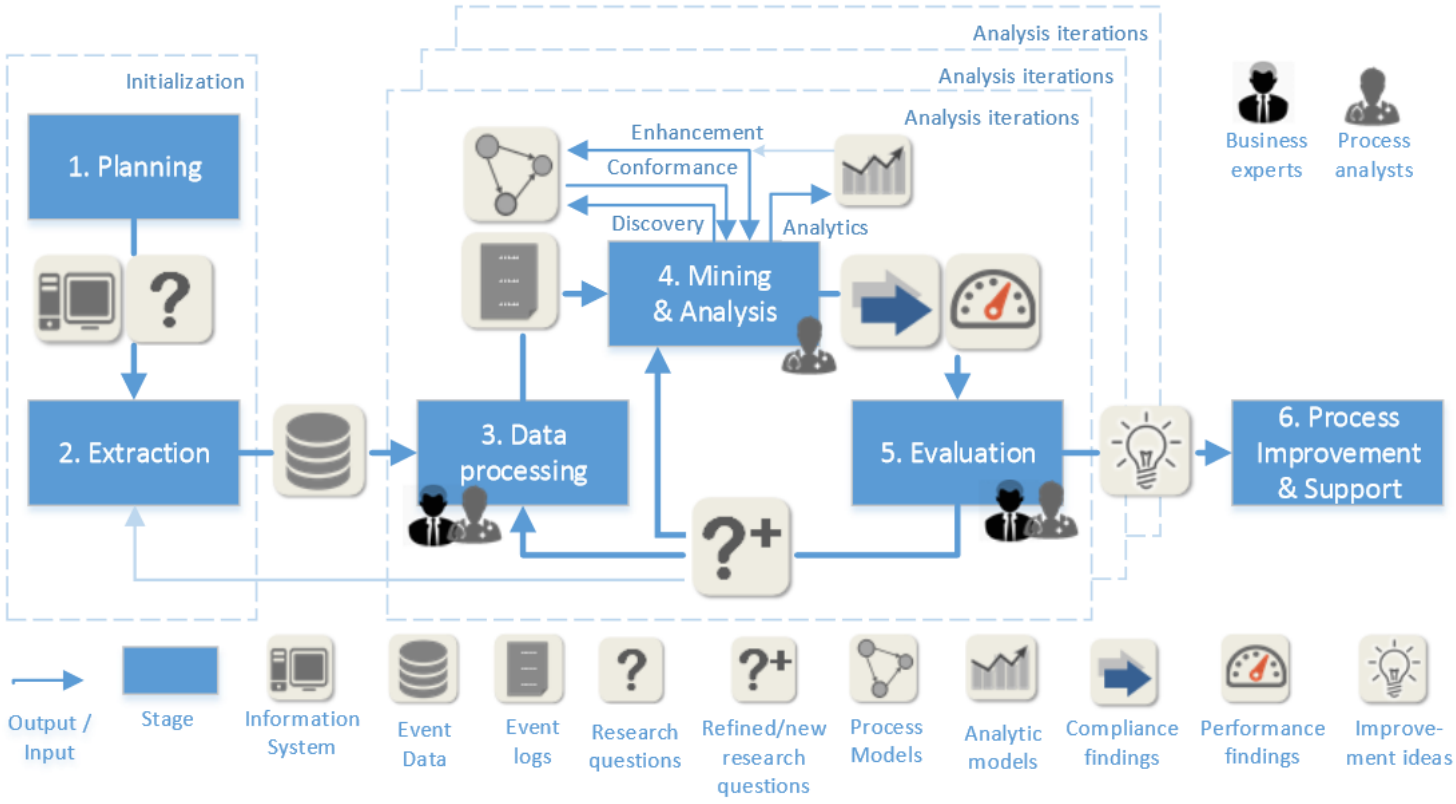
Process Mining Project

Christian-Albrechts-Universität zu Kiel

- Put it in an order…

Process Mining Project Methodology

Christian-Albrechts-Universität zu Kiel



M.L. Eck, van, X. Lu, S.J.J. Leemans, W.M.P. Aalst, van der:
PM2 : a Process Mining Project Methodology, CAiSE 2015, Springer

Process Mining Project Methodology

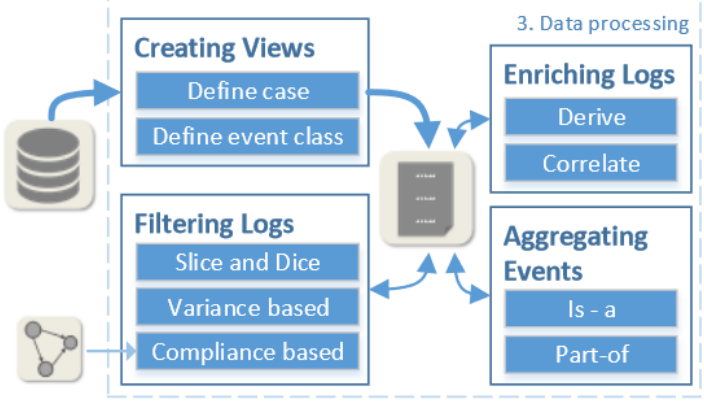
Christian-Albrechts-Universität zu Kiel

- **Stage 1:** Planning: set up the project and to determine the researchquestions:
  - Selecting business processes.
  - Identifying research questions
  - Composing project team

- **Stage 2:** Extraction: extract event data and, optionally, process models:
  - determining scope
  - extracting event data
  - transferring process knowledge

Process Mining Project Methodology

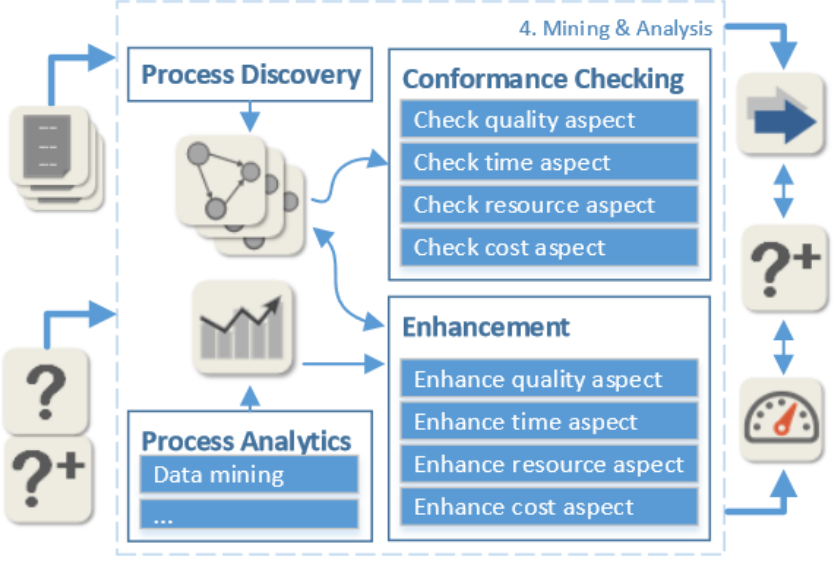
- **Stage 3:** Data Processing: to create event logs as different views of the obtained event data and to process event logs in such a way that it is optimal for the mining and analysis stage:
  - creating views
  - aggregating events
  - enriching logs
  - filtering logs



Process Mining Project Methodology

Christian-Albrechts-Universität zu Kiel

- **Stage 4:** Mining & Analysis: apply process mining techniques and gain insight into processes performance and compliance:
  - Process Discovery
  - Conformance Checking
  - Enhancement
  - Process Analytics



Process Mining Project Methodology

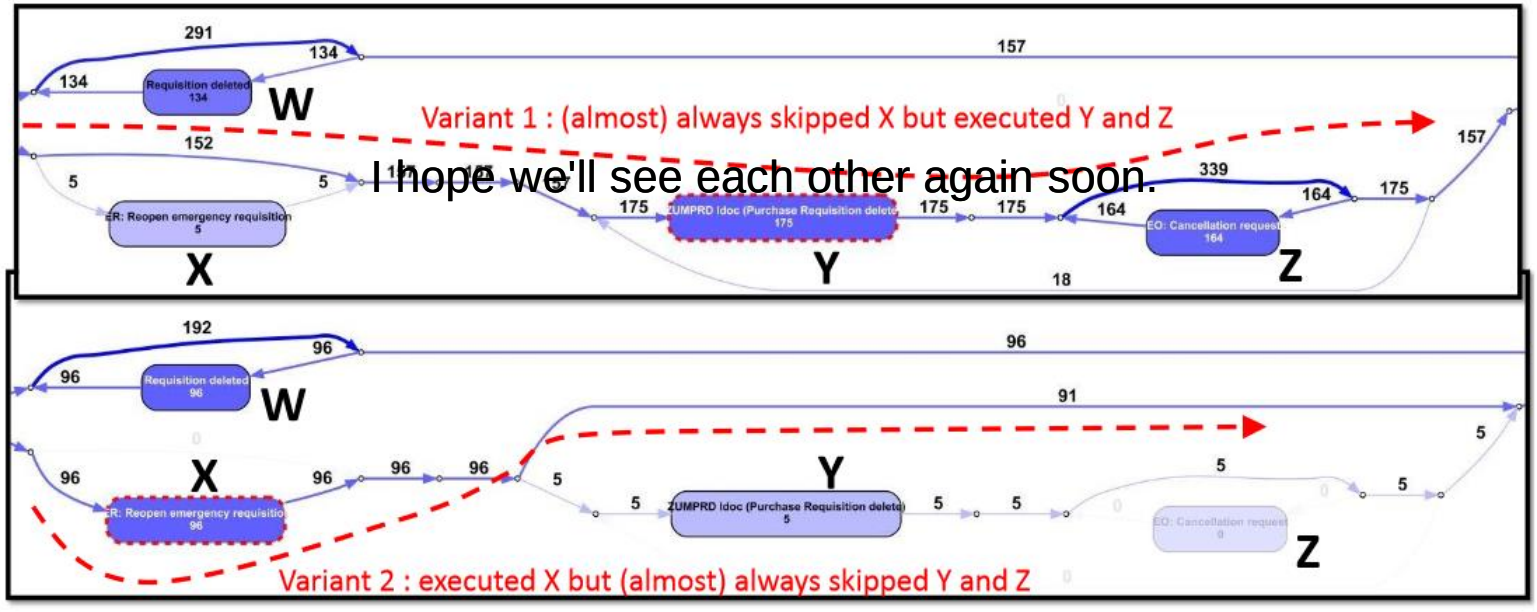
Christian-Albrechts-Universität zu Kiel

- **Stage 5:** Evaluation: to relate the analysis findings to improvement ideas that achieve the project's goals:
  - Diagnose
  - Verify & Validate

- **Stage 6:** Process Improvement & Support: to use the gained insights to modify the actual process execution:
  - Implementing improvements
  - Supporting operations

Two mined DFGs from the analysis

Christian-Albrechts-Universität zu Kiel



I hope we'll see each other again soon.

Process Mining
Directly-Follows Graph

Christian-Albrechts-Universität zu Kiel

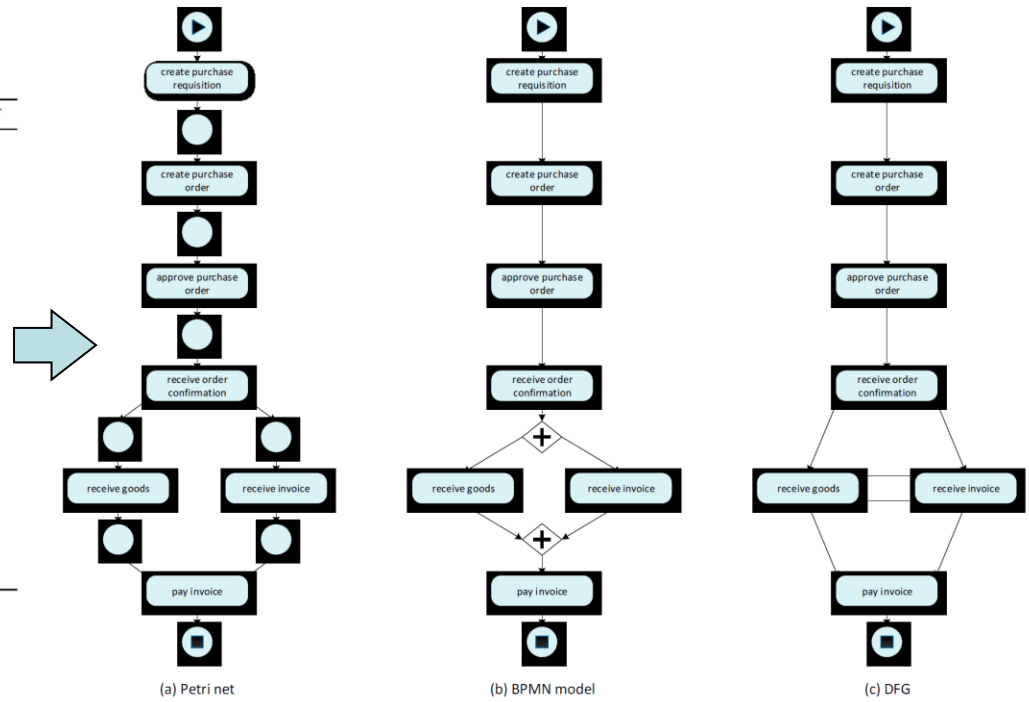
- Use event data to show what people, machines, and organizations are really doing
- Provides novel insights that can be used identify and address performance and compliance problems
- Commercial tools resort to producing Directly-Follows Graphs (DFGs) based on event data

Limitations of the Directly-Follows Graph

Christian-Albrechts-Universität zu Kiel

- to tackle complexity DFGs are seamlessly simplified by removing nodes and edges based on frequency thresholds
- DFGs may be misleading

case id (here an order)	activity	timestamp	resource	costs	customer
…	…	…	…	…	..
2019-88201	create purchase requisition	25-07-2019:09.15	John	€20.20	9950
2019-88201	create purchase order	25-07-2019:09.35	Mary	€48.30	9950
2019-88201	approve purchase order	25-07-2019:09.55	Sue	€30.70	9950
2019-88202	create purchase requisition	25-07-2019:10.15	John	€28.20	9955
2019-88202	create purchase order	25-07-2019:10.25	Mary	€29.30	9955
2019-88202	approve purchase order	25-07-2019:10.40	Sue	€37.60	9955
2019-88201	receive order confirmation	25-07-2019:11.50	Mary	€42.10	9950
2019-88201	receive goods	27-07-2019:09.35	Peter	€50.20	9950
2019-88202	receive order confirmation	27-07-2019:09.45	Mary	€42.30	9955
2019-88202	receive invoice	28-07-2019:10.10	Sue	€44.90	9955
2019-88201	receive invoice	28-07-2019:10.20	Sue	€30.80	9950
2019-88201	pay invoice	29-07-2019:11.05	Sue	€30.70	9950
2019-88202	receive goods	29-07-2019:11.35	Peter	€51.30	9955
2019-88202	pay invoice	29-07-2019:12.15	Sue	€29.20	9955
…	..	…	…	…	…



Possible Problems of DFG

- Activities that have a flexible ordering (e.g., due to concurrency) lead to Spaghetti-like DFGs with loops even when activities are executed at most once.
- DFGs can be simplified using frequency-based thresholds. However, this may lead to all kinds of interpretation problems due to "invisible gaps" in the model.
- Performance information mapped onto DFGs can be misleading, e.g., the average time reported between two activities is conditional (only the situations where they directly follow each other are considered).

Creating Directly-Follows Graph (DFG) (1)

- An a-event is an *event* that corresponds to activity a.

  A trace $\sigma = \langle a_1, a_2, a_3, \ldots, a_n \rangle$ is a sequence of activities.
- $\#_L(\sigma)$ is the number of cases in event log L that correspond to trace σ. Many cases may have the same trace.
- $\#_L(a)$ is the number of a-events in event log L.
- $\#_L(a, b)$ is the number of times an a-event is directly followed by a b-event within the same case. Without loss of generality, we assume that each case starts with a start event (denoted ▶) and end with an end event (denoted ■).
- If such start and end activities do not exist, they can be added to the start and end of each case. Traces are of the form $\sigma = \langle \blacktriangleright, a_2, a_3, \ldots, a_{n-1}, \blacksquare \rangle$

Creating Directly-Follows Graph (DFG) (2)

- A DFG is a graph with nodes that correspond to activities and directed edges that corresponds to directly-follows relationships.
- There are three parameters, i.e., τ_{var} , τ_{act} , and τ_{df} , that define thresholds for the minimal number of traces for each variant included (based on $\#_L(\sigma)$), the minimal number of events for each activity included (based on $\#_L(a)$), and the minimal number of direct successions for each relation included (based on $\#_L(a,b)$).
- Input: event log L and parameters τ_{var}, τ_{act}, and τ_{df}.
- Remove all cases from L having a trace with a frequency lower than τ_{var} , i.e., keep all cases with a trace σ such that $\#_L(\sigma) \geq \tau_{var}$. The new event log is L'. Note that the number of cases may have be reduced considerably, but the retained cases remain unchanged.

Creating Directly-Follows Graph (DFG) (2)

Input: event log L and parameters τ_{var}, τ_{act}, and τ_{df}.

1.Remove all cases from L having a trace with a frequency lower than τ_{var} , i.e., keep all cases with a trace σ such that $\#_L(\sigma) \geq \tau_{var}$. The new event log is L'. Note that the number of cases may have be reduced considerably, but the retained cases remain unchanged.

2.Remove all events from L' corresponding to activities with a frequency lower than τ_{act} , i.e., keep events for which the corresponding activity a meets the requirement $\#_{L'}(a) \geq \tau_{act}$. The new event log is L''. Note that the number of cases did not change, but the number of events may be much lower.

3.Add a node for each activity remaining in the filtered event log L''.

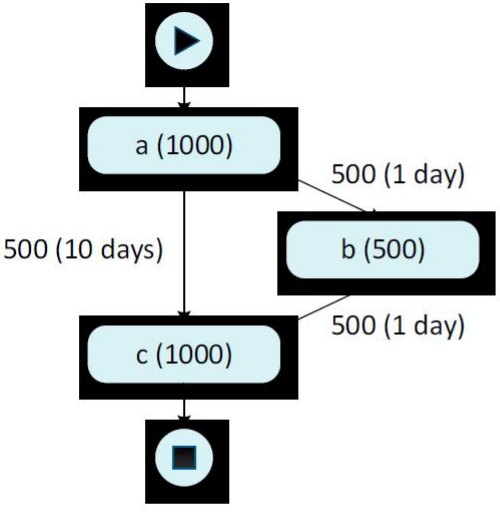
4.Connect the nodes that meet the τ_{df} threshold

5.Output the resulting graph.

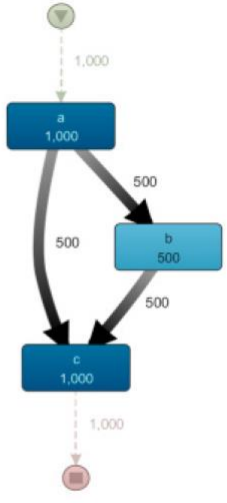
Creating Directly-Follows Graph (DFG) (4)

Christian-Albrechts-Universität zu Kiel

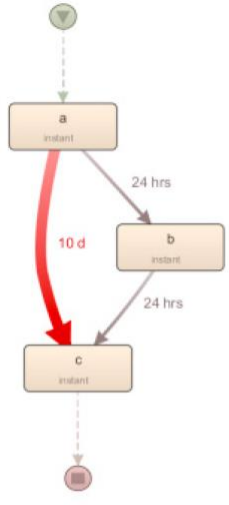
- Filtering edges based on the graph will miss that *a* is directly followed by *c* after removing *b*.



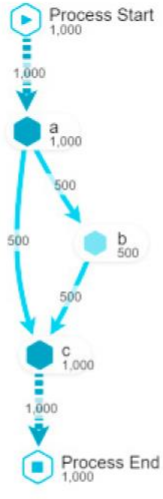
(a) DFG with all activities showing both frequencies and times



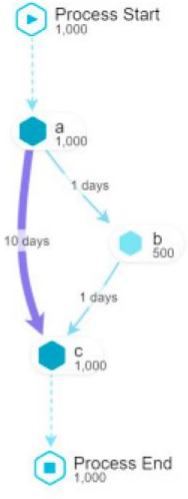
(b) DFG with all activities in Disco showing frequencies



(c) DFG with all activities in Disco showing times



(d) DFG with all activities in Celonis showing frequencies

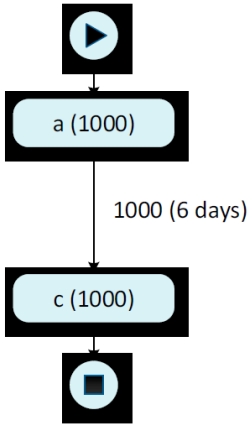


(e) DFG with all activities in Celonis showing times

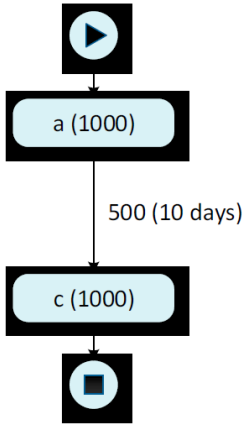
Differences between alternative DFG computations (1)

Christian-Albrechts-Universität zu Kiel

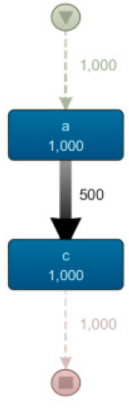
- $L = [\langle a, b, c \rangle^{500}, \langle a, c \rangle^{500}]$
- Event log L has 1000 cases and two variants:
  500 cases follow trace $\langle a, b, c \rangle$ and 500 cases follow trace $\langle a, c \rangle$.
- The time in-between activities a and b and the time in-between activities b and c in the first variant is always precisely one day (i.e., two days in total).



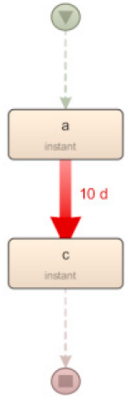
) correct DFG with only
he two most frequent
tivities (b was removed)



(g) incorrect DFG with only the two most frequent activities (b was removed)



(h) DFG generated by Disco for only the most frequent activities and showing frequencies

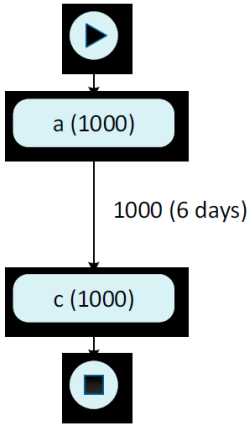


(i) DFG generated by Disco for only the most frequent activities and showing times

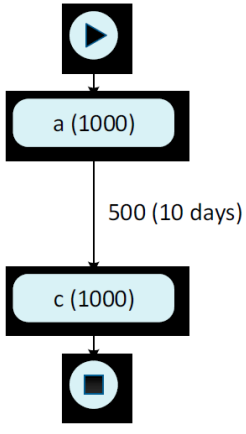
Differences between alternative DFG computations (2)

Christian-Albrechts-Universität zu Kiel

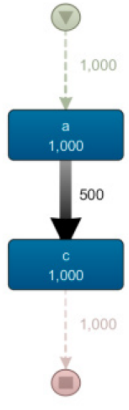
- However, also correctly generated DFGs have the problem that different interleavings of the same set of activities automatically leads to loops even when things are executed only once.
- The time in-between activities a and c in the second variant is always precisely 10 days.



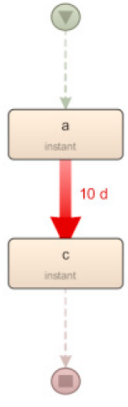
) correct DFG with only
he two most frequent
tivities (b was removed)



(g) incorrect DFG with only the two most frequent activities (b was removed)



(h) DFG generated by Disco for only the most frequent activities and showing frequencies



(i) DFG generated by Disco for only the most frequent activities and showing times

Event log with 10,000 cases (1)

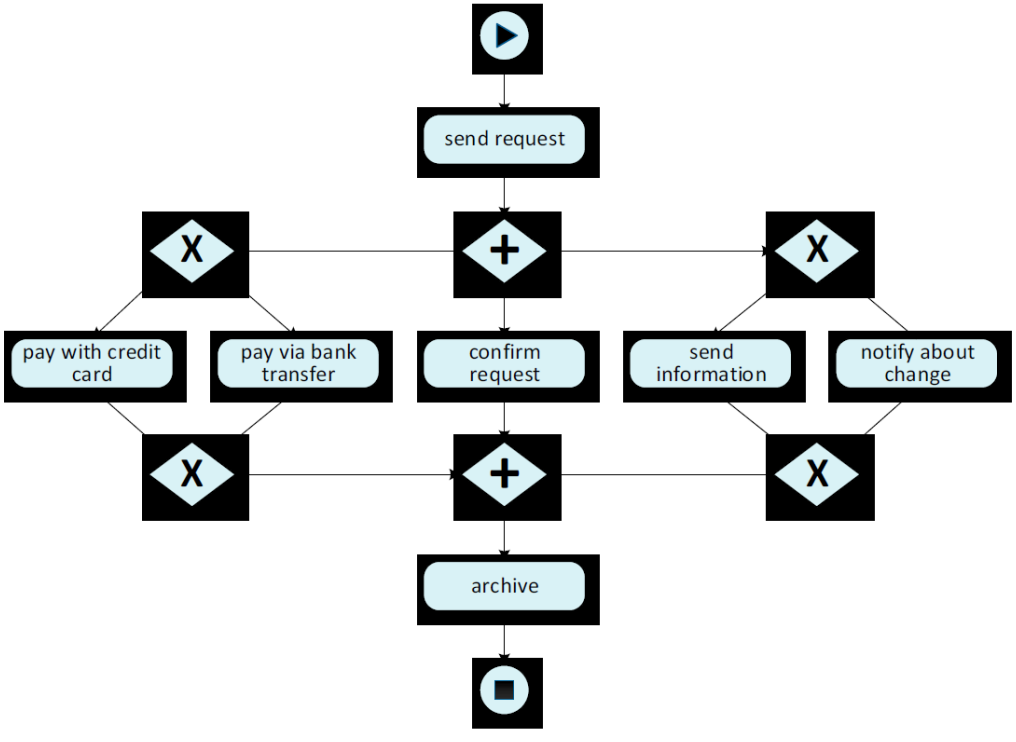
Christian-Albrechts-Universität zu Kiel

- The process starts with activity *send request* and ends with activity *archive*.
- Three independent parallel branches:

(1) a choice between activity *pay with credit card* and activity *pay via bank transfer*,

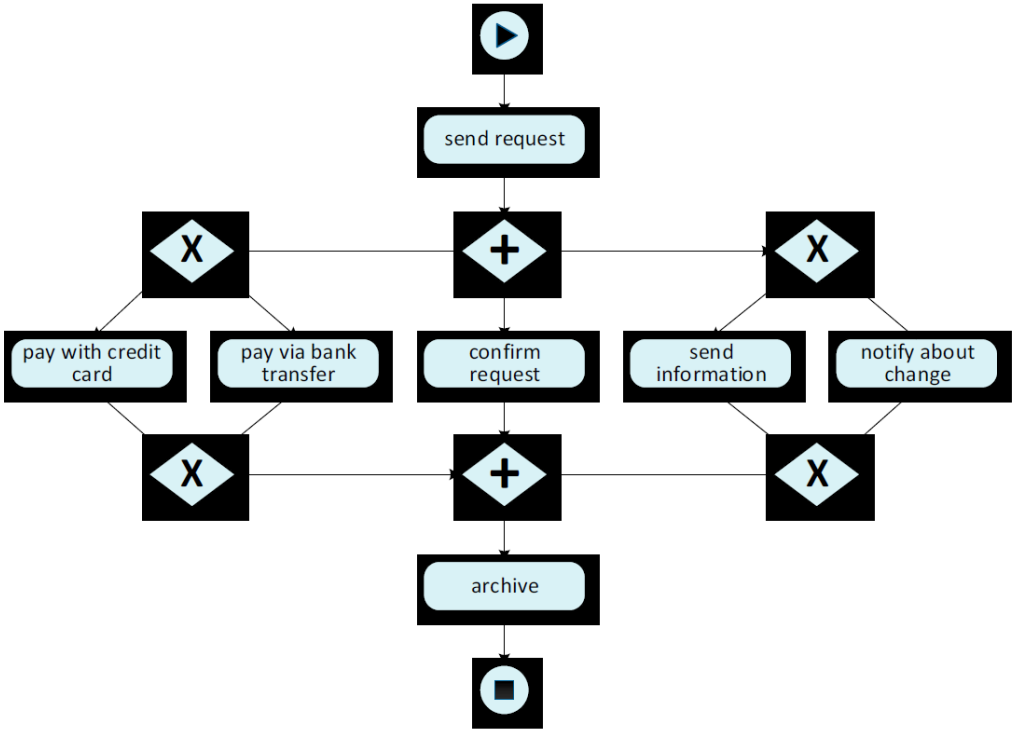
(2) activity *confirm request*, and

(3) a possible loop involving activities *send information* and *notify about change*



Event log with 10,000 cases (2)

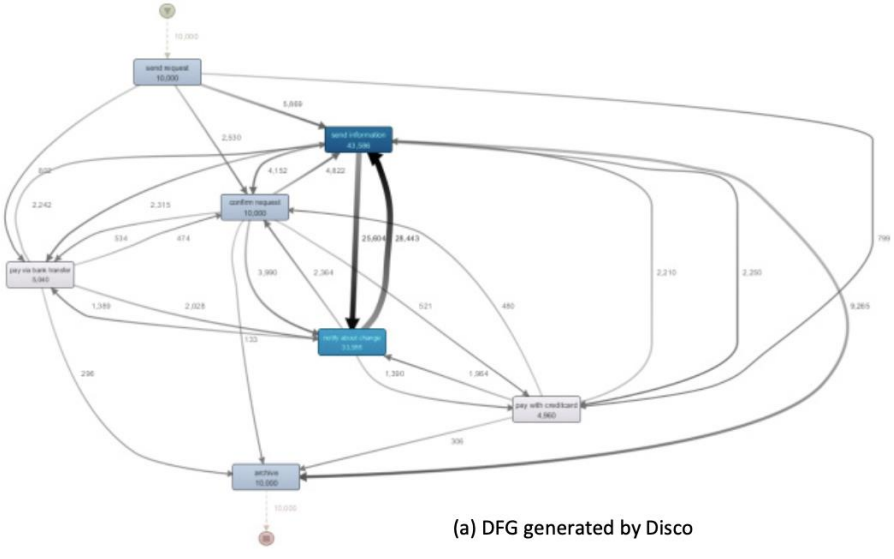
- CPN Tools to simulate the process and generated 10,000 cases following the process. In total, the event log has 117,172 events and 7 unique activities. The 10,000 cases correspond to 1159 process variants. The most frequent variant occurs 96 times. 30% of all variants occurred only once. 80% of the cases are described by 31% of the variants.



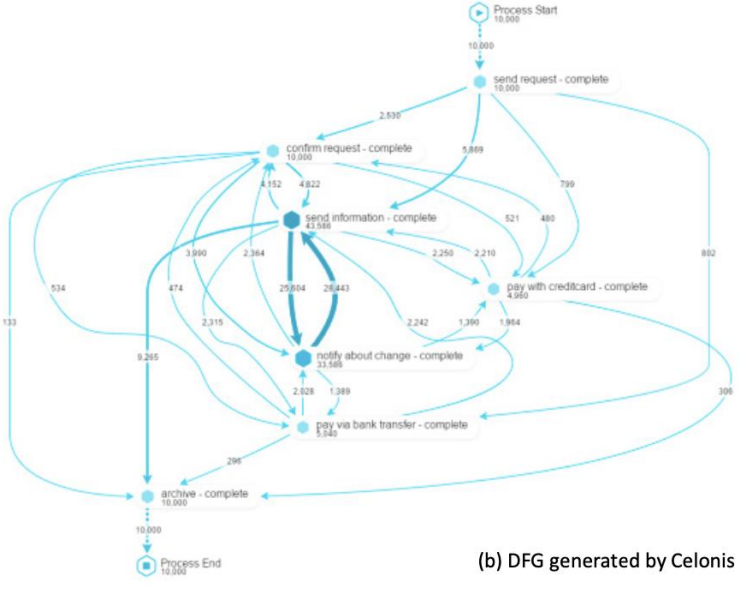
Difference between Disco and Celonis (1)

Christian-Albrechts-Universität zu Kiel

- One DFG was generated by Disco and the other DFG was generated by Celonis. The two DFGs are identical apart from their layout. Due to the different ways in which activities can be ordered, the DFG has many edges and these edges form loops also among activities that are executed at most once



(a) DFG generated by Disco

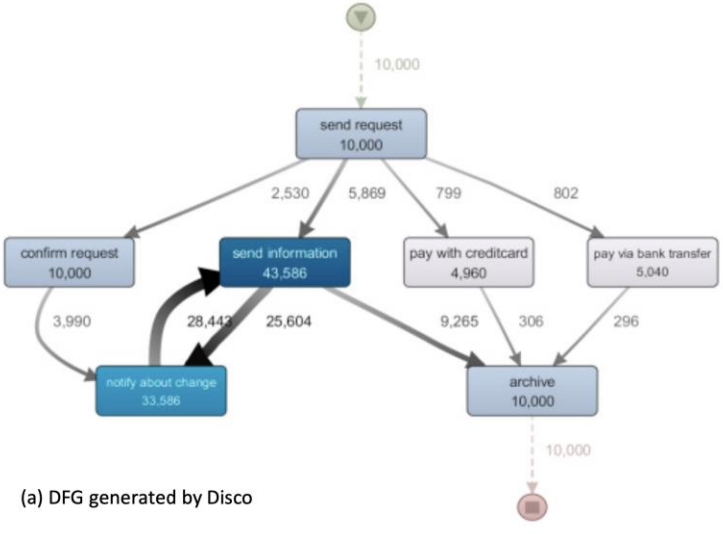


(b) DFG generated by Celonis

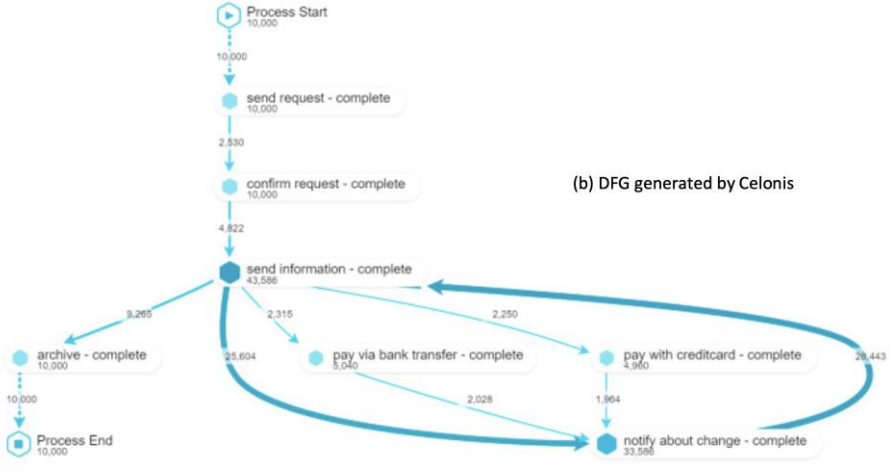
Difference between Disco and Celonis (2)

Christian-Albrechts-Universität zu Kiel

- It is also quite disturbing that two tools generate two completely different DFGs allowing for contradictory conclusions.
- For example, the average time between *send request* and *confirm request* is 3.5 days (set in simulation model), but Disco and Celonis both report 1.5 days (considering only 2,530 of 10,000 cases). Hence, one cannot rely on DFG-based performance diagnostics.



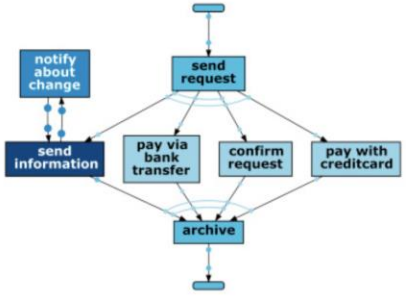
(a) DFG generated by Disco



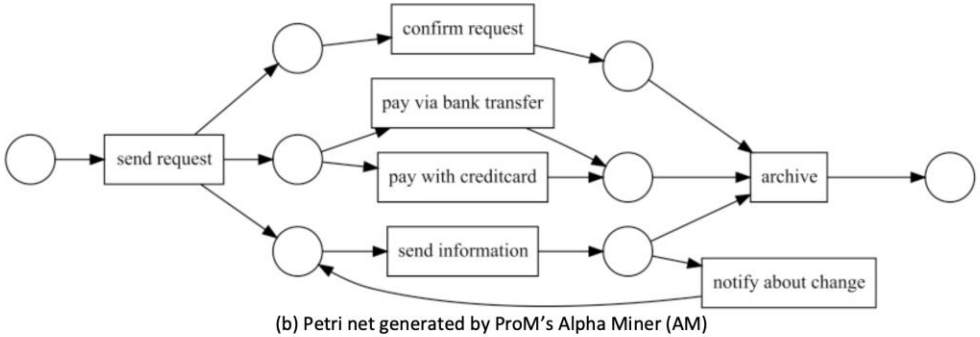
(b) DFG generated by Celonis

More different tools

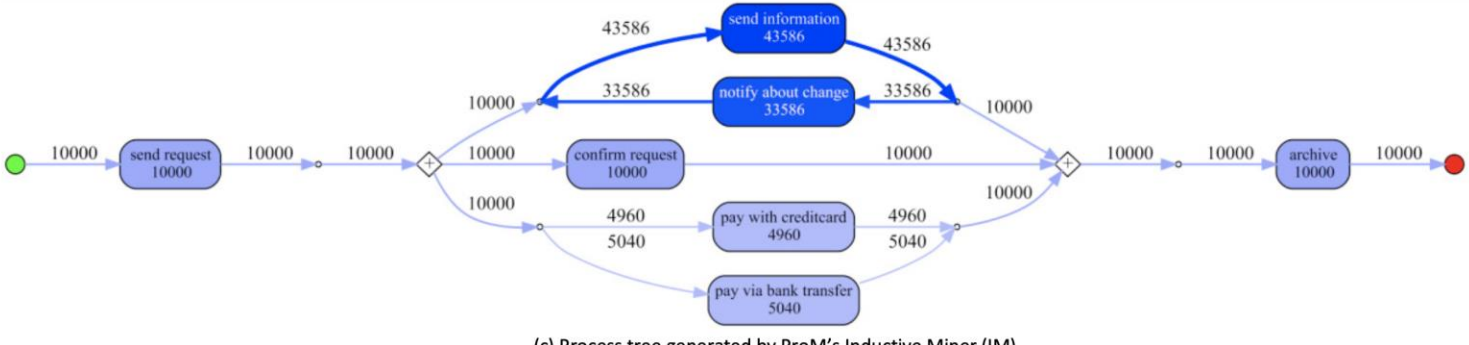
- The three process models were discovered using three different process discovery techniques implemented in ProM. All three models are behaviorally equivalent to the original process model that was used to generate the event log.



(a) C-net generated by ProM's Heuristic Miner (HM)



(b) Petri net generated by ProM's Alpha Miner (AM)

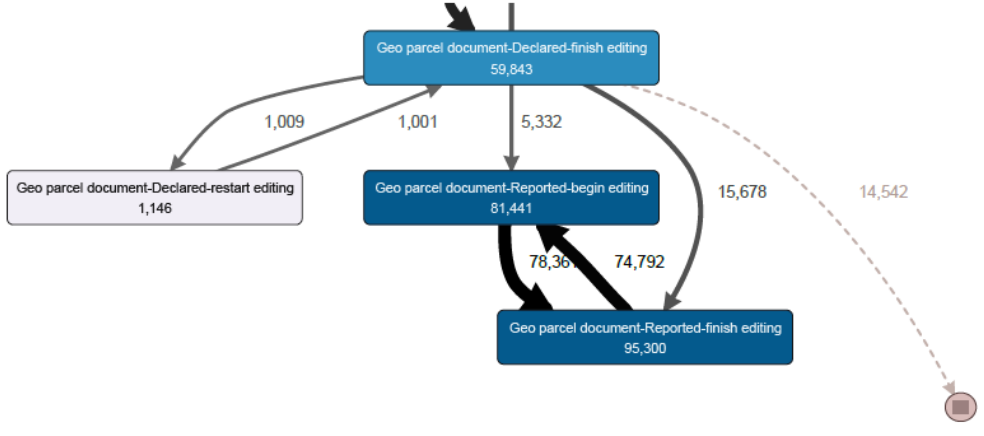


(c) Process tree generated by ProM's Inductive Miner (IM)

Soundness & DFG

Christian-Albrechts-Universität zu Kiel

Definition (Directly follows model - syntax). *Given an alphabet* Σ *such that* $start \notin \Sigma$ *and* $end \notin \Sigma$, *a* directly follows model *is a directed graph* (N, E), *such that* $N : \Sigma \cup \{start, end\}$ *is a set of nodes and* $E : N \times N$ *is a set of edges. Additionally,* $start$ *has no incoming and* end *has no outgoing edges:* $\neg\exists_n (n, start) \in E$ *and* $\neg\exists_n (end, n) \in E$.



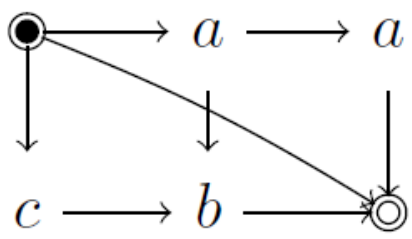
S. J.J. Leemans, E. Poppe, M. T. Wynn: Directly Follows-Based Process Mining: Exploration & a Case Study, ICPM 2019, IEEE

Soundness & DFG

- In the construction of DFG edges are filtered out based on a user chosen parameter to reduce the complexity of the model
  → this may lead to soundness issues

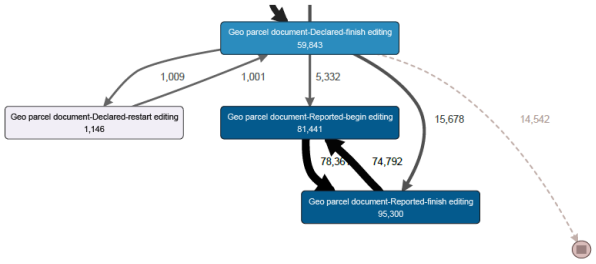
Definition (DFM soundness). *Let (N, E) be a DFM. Then, the DFM is* sound *if every node $\in N$ is on a path from start to end:*

$$\forall_{x \in N} \exists_{a_1 \ldots a_n \in N} a_1 = start \wedge a_n = end \wedge \exists a_j = x \\ \wedge \forall_{1 \leq i < n} (a_i, a_{i+1}) \in E$$



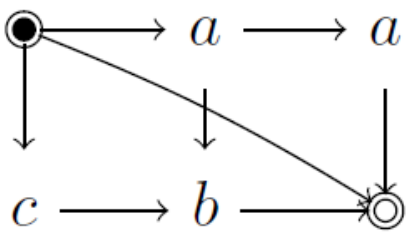
- The DFM without any edges is not sound, as the start and end nodes are not on a path as requested by the definition.

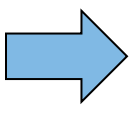
DFG translated to a Petri net

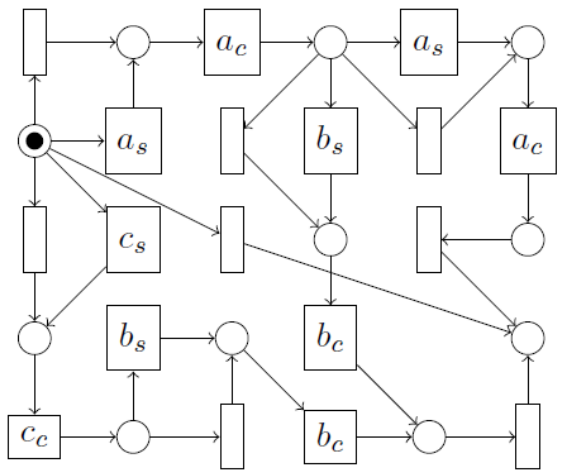


removing the least occurring edges might result in unsound DFMs

- A sound DFM can be translated to a language equivalent sound workflow net:







S. J.J. Leemans, E. Poppe, M. T. Wynn: Directly Follows-Based Process Mining: Exploration & a Case Study, ICPM 2019, IEEE

Directly Follows visual Miner

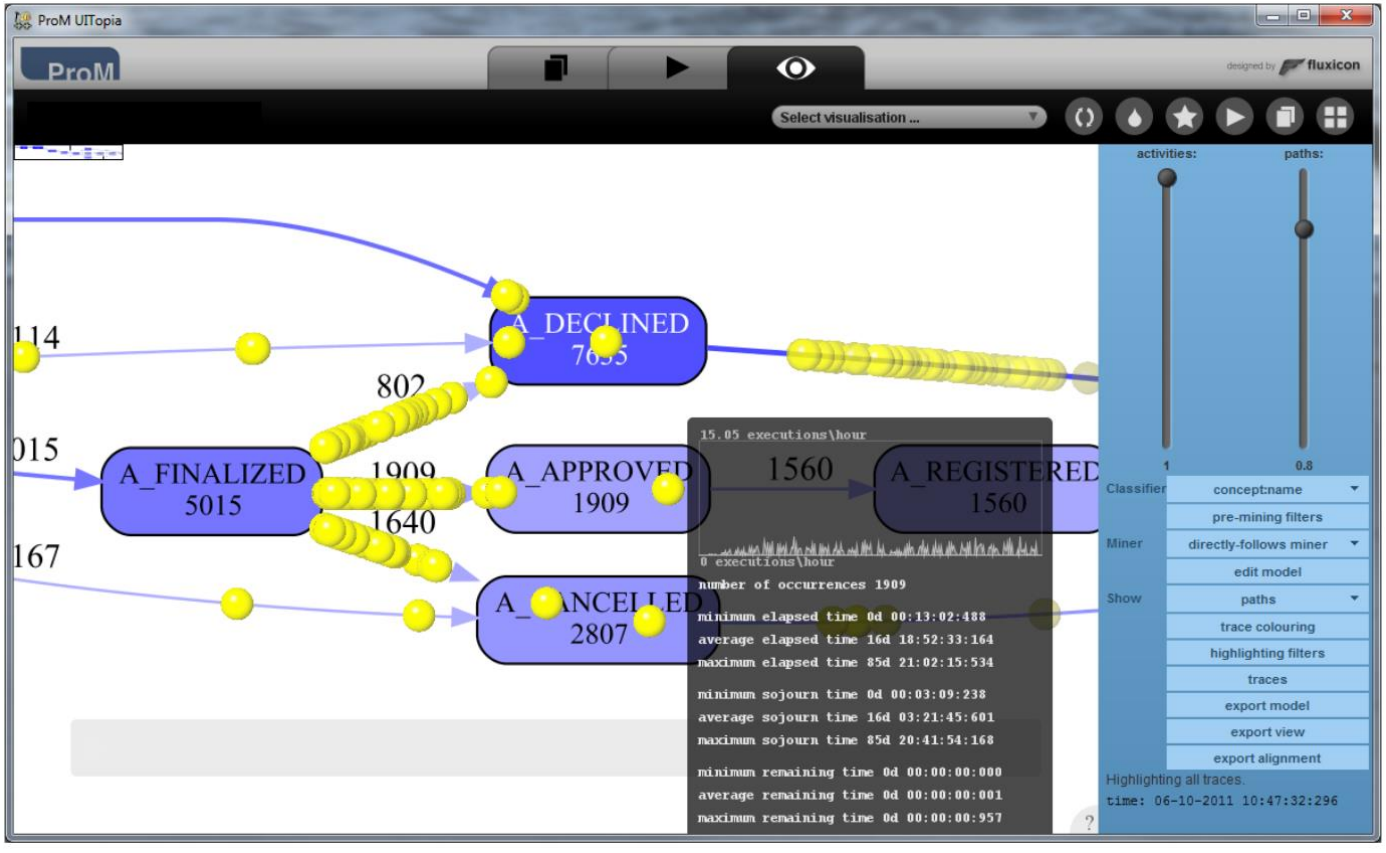
Christian-Albrechts-Universität zu Kiel

- takes as input an event log
- automatically discovers a directly follows process model
- applies conformance checking
- provides performance measure
- allows for filtering the log
- extension of the inductive visual miner

S. J.J. Leemans, E. Poppe, M. T. Wynn: Directly Follows-Based Process Mining: a Tool, ICPM 2019 Demo

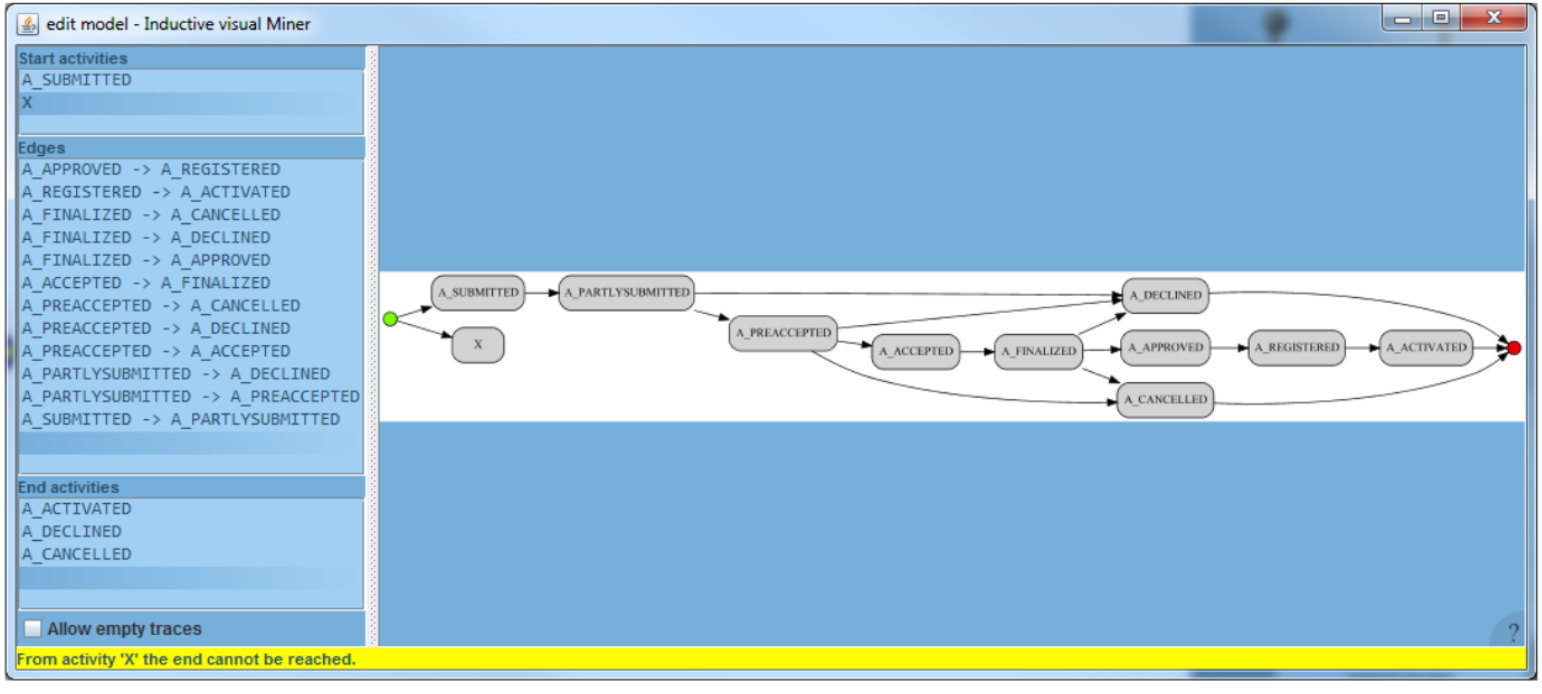
Directly Follows visual Miner

Christian-Albrechts-Universität zu Kiel



Directly Follows visual Miner

Christian-Albrechts-Universität zu Kiel



https://www.youtube.com/watch?v=xTKKqGwzh6I&feature=youtu.be

Chapter 2

Christian-Albrechts-Universität zu Kiel