Important Factors: Complexity Dimensions for Petri Nets

Patrizia Schalk[1]

[1]University of Augsburg, Universitätsstraße 6a, Augsburg, 86135, Germany

Abstract

Modeling languages like Petri nets are essential to visualize and analyze real-life processes. However, depending on the process at hand, these models can become fairly complex. Yet, complex models hinder users from understanding the underlying process. Thus, the literature suggests measures to quantify the complexity of a model in terms of a non-negative real number. The number of measures suggested in the literature is large, since there are many reasons why a model is complex. Due to this large number of measures, it is difficult to pick one for the evaluation of a model's complexity. In turn, previous research performed a factor analysis to group complexity measures that evaluate similar dimensions of complexity. However, this research focuses specifically on BPMN models instead of Petri nets. This paper therefore extends this line of research by performing a similar factor analysis on automatically discovered Petri nets. It analyzes a total of 14 complexity measures across 5 discovery algorithms. By performing one analysis per algorithm, it finds that the choice of the discovery algorithm influences the found complexity dimensions. The paper thus provides a publicly available tool to enable researchers and practitioners to repeat the analysis on other populations and with their preferred discovery algorithm.

Keywords

Process models, Petri nets, Complexity, Simplicity, Factor analysis

1. Introduction

Processes drive our daily lives and research. Systems engineering investigates processes to find how different parts of a software system behave and communicate [1]. Systems biology, on the other hand, analyzes them to understand chemical reactions and biological networks [2]. Finally, process mining studies them to find inefficient parts of a business process [3]. All of these disciplines create models to enable humans to learn more about the modeled process. In turn, it is of general interest to keep these models as simple as possible while retaining all important aspects of the modeled process.

Petri nets are one of the standard modeling languages used in all the aforementioned disciplines. Their simple structure and small set of semantic rules allows users to understand them quickly. At the same time, Petri nets feature a large array of formal foundations for automated analysis [4]. Fig. 1 shows two Petri net examples. Both of them model the same process. First, we choose whether to execute a, b, c, or d. Then, if we picked a or b, we execute e and g in parallel. If we picked c or d, we execute f and h in parallel instead. Finally, the process finishes with the execution of i.

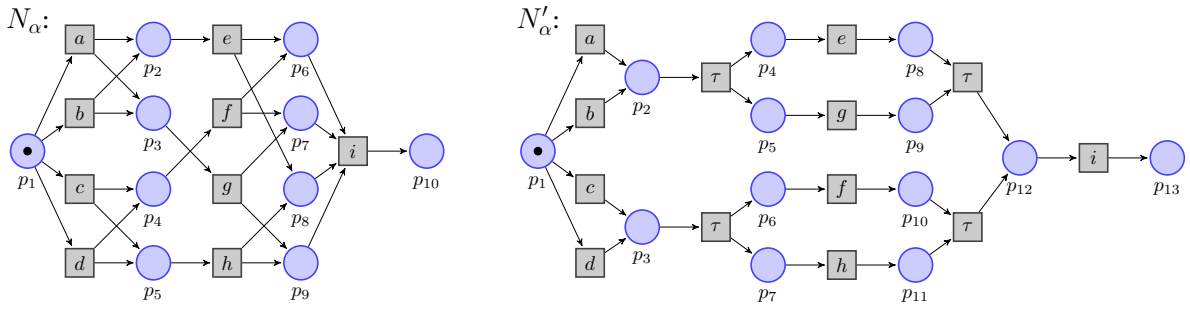


Figure 1: Two Petri net models, N_α and N'_α that model the same process but have different structure.

ATAED'26: Algorithms & Theories for the Analysis of Event Data 2026, June 23, 2026, Hamburg, GER

patrizia.schalk@uni-a.de (P. Schalk)

0009-0001-7757-6628 (P. Schalk)

The Petri net N_α of Fig. 1 models this process with fewer nodes than the net N'_α of the same figure. However, due to many edge crossings, the process is more difficult to identify in N_α than in N'_α. This example shows that the number of nodes is not the only important factor for a model's complexity. In turn, the literature suggests various measures to evaluate the complexity of a model. Most of these measures were originally defined for BPMN (Business Process Model and Notation) or EPC (Event Process Chains) [5]. However, since these measures typically count model elements, they feature natural translations to equivalent measures for Petri nets [6].

Due to the large amount of complexity measures, users often find it difficult to choose one for their application. Therefore, previous research performed a factor analysis on complexity measures for BPMN to find groups of measures that capture similar aspects of complexity [7]. This paper continues this line of research by performing a similar factor analysis on complexity measures for Petri nets. It finds that the chosen strategy to discover Petri nets influences the results of a factor analysis on their model complexity. The paper performs a factor analysis on 14 complexity measures for Petri nets using 5 different discovery strategies and makes the following contributions.

- A statistical analysis on whether the factors found by Lieben et al. [7] influence the complexity of automatically discovered Petri nets, for 14 complexity measures and 5 discovery algorithms; and
- A publicly available tool[1] to reproduce the results of this paper and to perform a factor analysis on different populations and with other discovery algorithms.

The remainder of this paper is structured as follows. Section 2 discusses related work and describes the factor analysis of Lieben et al. [7]. Section 3 then shows definitions for Petri nets and their model elements, before Section 4 introduces the investigated complexity measures. In Section 5, we describe and perform the factor analysis on these complexity measures separately for 5 discovery algorithms. The results of these factor analyses are discussed in Section 6. Afterward, Section 7 presents a tool to reproduce the factor analyses. Finally, Section 8 summarizes and gives directions for future research.

2. Related Work

Most model complexity measures in the literature evaluate BPMN or EPC models. Mendling [5] introduces 15 model complexity measures for EPC models and shows that they can predict modeling errors in the SAP reference model. Since EPC and BPMN models have similar syntax and semantics, the measures introduced by Mendling can also be used for BPMN models.

Lieben et al. [7] use the BPMN versions of these complexity measures alongside three additional measures [8, 9, 10] as variables for a factor analysis. As a sample, they use 4 150 hand-crafted models from a repository of the *Business Process Management Academic Initiative*. They filter out models with a connectedness of less than 50% and models with boundary events or disconnected activities. The authors justify this step by highlighting that they aim to analyze models that could have been returned by a discovery algorithm. To diversify the population, they also generate 300 process trees with the `PTandLogGenerator` [11] and convert them to BPMN models. They note that the latter set of models is biased in the sense that some complexity measures always return the same score for them. The authors state that this is not a problem, since the hand-crafted models do not share this bias.

For the analysis, they choose an R-type common factor analysis. After a scree test, they decide to set the number of factors to 4 and to ignore variables that have less than 30% of their variance explained by the factors. In this way, the authors find that the complexity measures fall into four dimensions.

- Token Behavior Complexity, which contains measures that are influenced by the flow of tokens;
- Node IO Complexity, which encompasses measures evaluating the degree of nodes in the model;
- Path Complexity, whose measures describe the complexity of the paths in the model; and
- Degree of Connectedness, which contains measures that describe how dense a model is.

The authors also find that three of the complexity measures do not fit into any of these dimensions. They conclude that users should pick one measure from each dimension, alongside one of these three unfitting measures, to evaluate the overall complexity of their models.

[1] Tool available at https://github.com/Pati-nets/ComFy

With their analysis, Lieben et al. [7] considerably reduce the number of complexity measures to choose from. However, the first dimension still contains seven different measures. Thus, Schalk et al. [6] expand the analysis by studying mathematical properties of each analyzed complexity measure. However, since literature shows a shift from BPMN to Petri nets, their analysis concerns translations of the measures to complexity measures for Petri nets. This opens a gap between the analyses of Lieben et al. [7] and Schalk et al. [6] that is not yet addressed in the literature. This paper aims to close this gap by repeating the factor analysis of Lieben et al. for the complexity measures investigated by Schalk et al.

3. Basic Definitions

We denote the set of natural numbers by $\mathbb{N} = \{1, 2, 3, \dots\}$ and the set of non-negative integers by $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Furthermore, we denote the set of non-negative real numbers by $\mathbb{R}_0^+$. Next, we define the syntax of the modeling language that we investigate in this paper.

Definition 1. ***((Labeled) Petri nets).*** *A Petri net is a triple $N = (P, T, F)$ where P denotes a finite set of places, T denotes a finite set of transitions, and $F \subseteq (P \times T) \cup (T \times P)$ is a flow relation. For a place $p \in P$, we define ${}^\bullet p := \{t \in T \mid (t, p) \in F\}$ as the preset of p, i.e., the set of transitions that have an outgoing arc to p. Similarly, we define $p^\bullet := \{t \in T \mid (p, t) \in F\}$ as the postset of p, i.e., the set of transitions that have an incoming arc from p. We define the preset and postset of a transition accordingly.*

A labeled Petri net is a 5-tuple $\mathcal{N} = (P, T, F, A, \ell)$, where (P, T, F) is a Petri net, A is an alphabet with $\tau \notin A$, and $\ell : T \to (A \cup \tau)$ is a labeling function assigning a (not necessarily unique) label to each transition $t \in T$. A transition $t \in T$ with $\ell(t) = \tau$ is called a silent transition.

Fig. 1 shows two Petri nets, N_α and N'_α. In each of these nets, places are drawn as circles and transitions as rectangles. We write transition labels inside these rectangles. Since all transitions in the net N_α have a unique label, we can also interpret it as an unlabeled Petri net. The net N'_α, on the other hand, contains four silent τ transitions. These transitions have no externally visible semantics; that is, every occurrence of τ in a word of a labeled Petri net's language is ignored.

Since we will only study complexity measures that investigate the structural properties of a Petri net, we will not formally define their semantics. Instead, we give an intuitive description. The language of a Petri net is defined by its structure and a *marking*, that is, a multiset of places, drawn as black bullets—called *tokens*—inside the places. A transition can fire if all places in its preset contain at least one token. If it fires, it consumes one token from each place in its preset and produces one token in each place of its postset. The language of a Petri net with respect to some fixed initial marking consists of all finite sequences of transitions that can fire one after another, starting at the initial marking. If a transition with the label τ occurs in such a sequence, it is omitted in the language of the Petri net.

In the net N_α of Fig. 1, the transitions a, b, c, and d can fire initially. Since they all share the same preset, they compete for the token in the place p_1 and only one of them can fire. We call such a structure an xor-split. If we decide to fire the transition a, it produces two tokens: One in the place p_2 and one in the place p_3. This enables the transitions e and g at the same time. Since their presets are disjoint, they can fire concurrently. Thus, we call a transition that initiates such concurrency an and-split. Ideally split nodes are countered by a corresponding join node, as defined in the following.

Definition 2. ***(Connectors in a Petri net).*** *Let $N = (P, T, F)$ be a Petri net. We define:*

- $\mathcal{S}^N_{xor} = \{p \in P \mid p^\bullet > 1\}$ *as the set of* xor*-split nodes,*
- $\mathcal{J}^N_{xor} = \{p \in P \mid {}^\bullet p > 1\}$ *as the set of* xor*-join nodes,*
- $\mathcal{S}^N_{and} = \{t \in T \mid t^\bullet > 1\}$ *as the set of* and*-split nodes, and*
- $\mathcal{J}^N_{and} = \{t \in T \mid {}^\bullet t > 1\}$ *as the set of* and*-join nodes.*

Furthermore, we denote by $\mathcal{C}^N_{xor} = \mathcal{S}^N_{xor} \cup \mathcal{J}^N_{xor}$ the set of xor*-connectors and by $\mathcal{C}^N_{and} = \mathcal{S}^N_{and} \cup \mathcal{J}^N_{and}$ the set of* and*-connectors. The set of all connectors is denoted by $\mathcal{C}^N = \mathcal{C}^N_{xor} \cup \mathcal{C}^N_{and}$.*

Note that $\mathcal{S}^N_{\text{xor}} \cap \mathcal{J}^N_{\text{xor}} \neq \emptyset$ or $\mathcal{S}^N_{\text{and}} \cap \mathcal{J}^N_{\text{and}} \neq \emptyset$ for some Petri nets N. Since none of the investigated measures relate the number of splits or joins to the number of connectors, we accept this property.

4. Complexity Measures

Let $\mathcal{M}$ be the set of all Petri nets. A complexity measure is a function $C : \mathcal{M} \to \mathbb{R}_0^+$ that assigns a real-valued complexity score to any Petri net. The higher the returned value, the more complex the input model. Table 1 reports the formal definitions of the complexity measures we analyze in this paper.

Table 1
Definition of the complexity measures analyzed in this paper for a (labeled) Petri net $N = (P, T, F, A, \ell)$.

	Measure	Definition	References
1.	$C_{\text{size}}(N)$	$\lvert P\rvert + \lvert T\rvert$	[5, 6]
2.	$C_{\text{MM}}(N)$	$\left\lvert \sum_{t \in \mathcal{S}^N_{\text{and}}} \lvert t^\bullet\rvert - \sum_{t \in \mathcal{J}^N_{\text{and}}} \lvert {}^\bullet t\rvert \right\rvert + \left\lvert \sum_{p \in \mathcal{S}^N_{\text{xor}}} \lvert p^\bullet\rvert - \sum_{p \in \mathcal{J}^N_{\text{xor}}} \lvert {}^\bullet p\rvert \right\rvert$	[5, 6]
3.	$C_{\text{CH}}(N)$	$-\left(\frac{\lvert\mathcal{C}^N_{\text{and}}\rvert}{\lvert\mathcal{C}^N\rvert} \cdot \log_2\left(\frac{\lvert\mathcal{C}^N_{\text{and}}\rvert}{\lvert\mathcal{C}^N\rvert}\right) + \frac{\lvert\mathcal{C}^N_{\text{xor}}\rvert}{\lvert\mathcal{C}^N\rvert} \cdot \log_2\left(\frac{\lvert\mathcal{C}^N_{\text{xor}}\rvert}{\lvert\mathcal{C}^N\rvert}\right)\right)$	[5, 6]
4.	$C_{\text{ts}}(N)$	$\sum_{t \in T}(\lvert t^\bullet\rvert - 1)$	[5, 6]
5.	$C_{\text{CFC}}(N)$	$\lvert\mathcal{S}^N_{\text{and}}\rvert + \sum_{p \in \mathcal{S}^N_{\text{xor}}} \lvert p^\bullet\rvert$	[5, 6]
6.	$C_{\text{sep}}(N)$	$1 - \frac{1}{\lvert P\rvert+\lvert T\rvert} \cdot \lvert\{v \in P \cup T \mid v \text{ is a cut-vertex in } N\}\rvert$	[5, 6]
7.	$C_{\text{acd}}(N)$	$\frac{1}{\lvert\mathcal{C}^N\rvert} \cdot \sum_{x \in \mathcal{C}^N}(\lvert {}^\bullet x\rvert + \lvert x^\bullet\rvert)$	[5, 6]
8.	$C_{\text{mcd}}(N)$	$\max\{\lvert {}^\bullet x\rvert + \lvert x^\bullet\rvert \mid x \in \mathcal{C}^N\}$	[5, 6]
9.	$C_{\text{seq}}(N)$	$1 - \frac{1}{\lvert F\rvert} \cdot \lvert\{(x, y) \in F \mid x, y \notin \mathcal{C}^N\}\rvert$	[5, 6]
10.	$C_{\text{CNC}}(N)$	$\frac{\lvert F\rvert}{\lvert P\rvert+\lvert T\rvert}$	[5, 6]
11.	$C_{\text{dens}}(N)$	$\frac{\lvert F\rvert}{2 \cdot \lvert T\rvert \cdot \lvert P\rvert}$	[5, 6]
12.	$C_{\text{dup}}(N)$	$\sum_{a \in A}(\max(\lvert\{t \in T \mid \ell(t) = a\}\rvert, 1) - 1)$	[6, 8]
13.	$C^\tau_{\text{dup}}(N)$	$\sum_{x \in A \cup \{\tau\}}(\max(\lvert\{t \in T \mid \ell(t) = x\}\rvert, 1) - 1)$	[6, 8]
14.	$C_\emptyset(N)$	$\lvert\{p \in P \mid {}^\bullet p \subseteq \mathcal{S}^N_{\text{and}} \wedge p^\bullet \subseteq \mathcal{J}^N_{\text{and}}\}\rvert$	[6, 9]

Most complexity measures count the number of complex structures in a Petri net. As such, the measure *size* C_{size} counts the number of places and transitions in a net. Similarly, the measure *connector mismatch* C_{MM} counts the differences between edges exiting splits and edges entering joins of each connector type. *Connector heterogeneity* C_{CH}, on the other hand, returns the entropy of connector types: A value between 0 and 1, where values close to 0 indicate that there is mostly one connector type in the net, while values close to 1 indicate that the net contains equally many connectors of each type. The measure *token split* C_{ts} counts how many edges must be deleted from the net to remove all parallelism and thus increases when many transitions produce more than one token. The *control flow complexity* C_{CFC} aims to count the number of possible control flows in the net. For each and-split, it infers a cost of one. For each xor-split, it punishes the complexity score by the number of edges exiting the split. The *separability* C_{sep}, on the other hand, evaluates how loosely the net is connected. It counts all vertices whose removal would increase the number of connected components in the net (i.e., its *cut-vertices*) and divides this number by the total amount of nodes. Since loosely connected Petri nets are easier to understand than heavily connected ones, the measure subtracts the resulting ratio from 1. The *average* and *maximum* connector degree C_{acd} and C_{mcd} determine the average and maximum number of incoming and outgoing edges of connector nodes, respectively. Thus, these measures punish Petri nets whose connectors have large pre- or postsets. Conversely, the *sequentiality* C_{seq} counts how many edges in the net connect non-connector nodes and relates this number to the total number of edges. Since a net is easier to understand if it contains many sequential arcs, this measure subtracts the resulting ratio from 1. The *coefficient of network connectivity* C_{CNC} and *density* C_{dens} count the number of edges and relate them to the number of nodes or the maximum possible number of edges, respectively. The measure *duplicate number of tasks* counts the number of label repetitions within the Petri net. For this measure, there are arguments for and against counting repetitions of silent labels. Counting duplicate τ labels allows the identification of unnecessary model structures consisting of only silent transitions. However, one could argue that multiple silent transitions should not be punished if

they improve the structure of the connector nodes. Thus, we introduce two versions of this measure: C_{dup} does not count duplicate silent transitions, while C^{τ}_{dup} does. Finally, the *number of empty sequence flows* $C_{\emptyset}$ counts the number of places with only and-splits in their preset and only and-joins in their postset. The idea is that such places do not restrict the behavior of the net and can thus be left out.

To see some examples, consider again the Petri nets N_α and N'_α of Fig. 1. Table 2 reports the scores that the measures in Table 1 return. If we compare the complexity scores of N_α and N'_α for each complexity measure, it becomes evident that there is no unique answer to which model is less complex: Some measures deem N_α to be more complex, while others rate N'_α as more complex. Furthermore, the scores of two different measures are incomparable, as each measure follows its own arbitrary scale: C_{size} can return any natural number, while C_{CH} returns only values between 0 and 1.

Table 2
Complexity scores for the exemplary Petri nets of Fig. 1.

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	C_{dup}	C^{τ}_{dup}	$C_{\emptyset}$
N_α	19	24	1	8	12	0.95	3.17	5	1	1.53	0.16	0	0	4
N'_α	26	2	1	2	6	0.92	3.13	4	0.7	1.15	0.09	0	3	0

Note that the literature contains more complexity measures than those reported in Table 1 [5, 10, 12]. For better comparability with the results by Lieben et al. [7], we restrict the analysis to the complexity measures investigated by these authors, with some exceptions. Lieben et al. include the measures *cross-connectivity*, *diameter*, *depth*, *cyclicity*, and *structuredness* in their analysis, which we omit here. The reason is that these five measures need non-linear runtime to compute complexity scores. This is not a problem in the setting of Lieben et al., where most models were hand-crafted and thus rather small. However, in our setting, we discover Petri nets automatically from large event logs. In turn, we expect model sizes in the several hundreds. The aforementioned complexity measures would need several hours to compute complexity scores per model, which is not feasible for large sample sizes. If the reader is interested in the analysis of these five measures, we refer to the tool described in Section 7.

5. Factor Analysis of Model Complexity

We start this section with general remarks on the performed factor analysis. The goal of the analysis is to find the underlying factors that influence the Petri net complexity measures reported in Table 1. To ensure comparability of our results with those by Lieben et al. [7], we use the same parameters for the factor analysis as them. However, we still argue why these parameters are fitting for our analysis.

As discussed in Section 4, we express the complexity of a Petri net with a non-negative real number. Hair et al. [13, p. 6–7] label datasets with this property *metric ratio scales*. In contrast to other scales (e.g., scales resulting from binary variables), metric ratio scales guarantee that correlation values can be calculated. Thus, the data returned by complexity measures is appropriate to perform a factor analysis.

We now need to decide on the type of factor analysis. Essentially, we have two choices: With an *R factor analysis*, we study the variables and identify dimensions that group them. On the other hand, with a *Q factor analysis*, we study the population that generated the data and identify groups among them. In our setting, a Q factor analysis would find groups of similar models in our population. Since we are interested in finding dimensions for complexity measures, we thus choose an R factor analysis.

Next, we have to decide whether to perform a *component factor analysis* or a *common factor analysis*. A component factor analysis considers the total variance of the variables, while a common factor analysis excludes the specific variance and error variance of each variable. In most applications, both types of factor analyses arrive at similar results [13, p. 108]. However, since we have no prior knowledge about the specific and error variance of each variable, we use a common factor analysis.

We now describe the analyzed population and its size. Hair et al. [13, p.102] propose to collect at least 10 data points per analyzed variable. Other researchers, such as Lieben et al. [7], collect at least 20 data points per variable. We thu choose a population size of at least $14 \cdot 20 = 280$. We perform the same

statistical tests on the variables' intercorrelations as Lieben et al. Since these tests return better scores for larger sample sizes [13, p. 104], we keep the population size as close to this lower bound as possible.

To ensure reproducibility of our results, we use publicly available data for our analysis. We thus decide to use the 288 training logs from the process discovery contest 2025 [14]. These logs describe complex processes that will likely result in complex model structures. To collect the necessary complexity data, we first fix a discovery algorithm and then execute it on these 288 event logs to obtain 288 Petri net models. We then calculate the complexity scores of these models using the 14 complexity measures reported in Table 1. In this way, we receive a population size that is large enough for a meaningful factor analysis and small enough to avoid effects where statistical tests succeed due to a large population size.

After collecting the data for the analysis, we perform two statistical tests [13, p. 104]. First, we execute *Bartlett's test of sphericity* (BToS). This test constructs a correlation matrix for the variables and returns whether this matrix features significant correlations among some of the variables. It expresses this in terms of a p-value. If this p-value is lower than 0.05, we consider the test successful. Second, we calculate the *measure of sampling adequacy* (MSA). This measure returns values between 0 and 1, where 1 means that all variables can be perfectly predicted by the other variables. The *variable-specific MSA* performs the same analysis for a fixed variable. Following the suggestion of Lieben et al. [7], we calculate all MSA values and consider the test successful if all of them are above 0.6. We exclude all variables with MSA values below 0.6 from the analysis and repeat the test on the remaining variables [13, p.104].

If both of these tests succeed, our data contains enough intercorrelation to perform a meaningful factor analysis. We then use the `Python` library `factor-analyzer` [15] to execute the factor analysis. As suggested by Lieben et al. [7], we set the number of factors to 4. This results in a matrix of factor loadings that indicate the effect of each factor on each variable. To distribute the variables among the factors evenly, it is common practice to perform a *rotation* to these factor loadings [13, p. 113]. Rotations facilitate the interpretation of factor loadings by rotating the underlying coordinate system, as depicted in Fig. 2. There are two main types of rotations: *Orthogonal rotations* and *oblique rotations*. Orthogonal rotations assume that the factors that define the coordinate system are linearly independent of each other. This means changing the value of one factor has no effect on the values of other factors. Since this is not necessarily realistic in all applications, oblique rotations allow to rotate the axes representing each factor independently. Following the analysis by Lieben et al. [7], we choose to perform a *varimax* rotation on all factor loadings. Note that varimax is an orthogonal rotation technique.

With the factor loadings of all variables calculated, we can determine the *communality* of each variable. A variable's communality is the ratio of its variance that can be explained by the found factors. Hair et al. [13, p. 119] propose to exclude variables with communalities lower than 0.5 from the interpretation of the results. However, Lieben et al. [7] exclude variables only if their communalities fall below 0.3. We will accept the latter bound to keep our analyses comparable to those by Lieben et al. However, we will always show the communalities of each variable, so the reader can ignore the interpretation of variables whose communalities they deem too low.

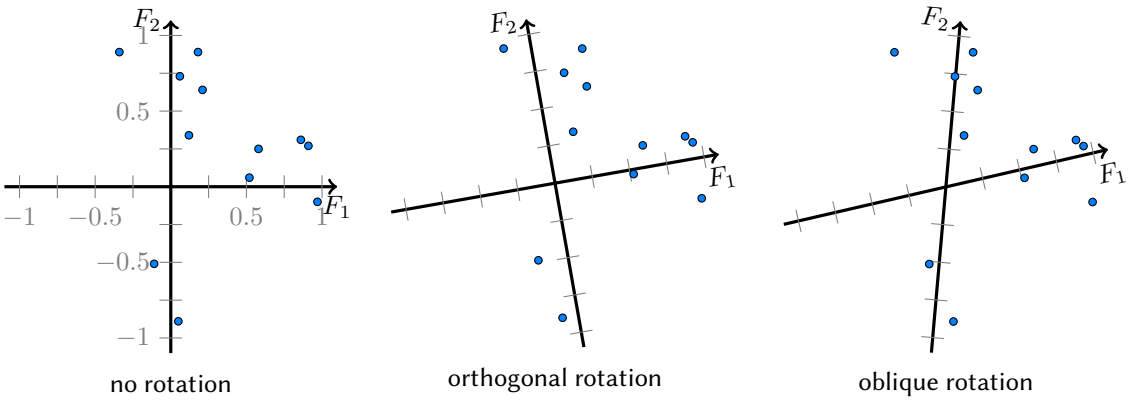


Figure 2: Examples for orthogonal and oblique rotations on the loadings of two factors, F_1 and F_2.

In the remainder of this section, we show the results of a factor analysis where models were discovered using a fixed algorithm. We use 5 discovery algorithms: The Alpha Miner [16], the Directly Follows Miner [17], the Inductive Miner [18], the Heuristics Miner [19], and the Hybrid ILP Miner [20]. We choose these discovery algorithms, since they feature a publicly available implementation in ProM [21] that executes the algorithms exactly as envisioned by their original authors. We compare the dimensions that we find in our analyses to those found by Lieben et al. [7] for complexity measures of BPMN models. The complexity dimensions they found are reported in Table 3.

Table 3
The resulting dimensions of the factor analysis by Lieben et al. [7]. Note that they used measures for BPMN models, corresponding to those in Table 1. Colored in gray are measures that we exclude in our analysis.

Dimension 1	Dimension 2	Dimension 3	Dimension 4	Excluded
$C^{\text{BPMN}}_{\text{size}}$, $C^{\text{BPMN}}_{\text{ts}}$, $C^{\text{BPMN}}_{\text{CFC}}$, $C^{\text{BPMN}}_{\text{MM}}$, $C^{\text{BPMN}}_{\text{CH}}$, $C^{\text{BPMN}}_{\text{sep}}$, $C^{\text{BPMN}}_{\text{CC}}$	$C^{\text{BPMN}}_{\text{CNC}}$, $C^{\text{BPMN}}_{\text{acd}}$, $C^{\text{BPMN}}_{\text{mcd}}$, $C^{\text{BPMN}}_{\text{seq}}$	$C^{\text{BPMN}}_{\text{cyc}}$, $C^{\text{BPMN}}_{\text{diam}}$, $C^{\text{BPMN}}_{\text{depth}}$	$C^{\text{BPMN}}_{\text{dens}}$, $C^{\text{BPMN}}_{\text{CNC}}$	$C^{\text{BPMN}}_{\emptyset}$, $C^{\text{BPMN}}_{\text{dup}}$, $C^{\text{BPMN}}_{\text{struct}}$

5.1. Alpha Miner

The Alpha Miner [16] is considered to be one of the first process discovery algorithms in the literature. It discovers a Petri net by first creating a transition for each event in the process. For two events, e_1 and e_2, it then investigates which of the following is true according to the data recorded for the process:

- e_1 always appears directly after e_2 in the data;
- e_2 always appears directly after e_1 in the data;
- e_1 sometimes appears directly after e_2 and e_2 sometimes appears directly after e_1; or
- e_1 and e_2 never appear directly next to each other in the data.

It then creates places and connections to transitions that allow the net to imitate these relations. We use the Alpha Miner to discover Petri nets for all 288 event logs in our population. After evaluating the complexity of these models using the measures reported in Table 1, we obtain 288 observations for each variable of our analysis. The descriptive statistics of these observations are reported in Table 4.

As highlighted in this table, the measures C_{dup} and C^{τ}_{dup} always return the value 0 for the models in our sample. However, $C_{\text{dup}}(N) = 0 = C^{\tau}_{\text{dup}}(N)$ is true for any model N discovered by the Alpha Miner, since the discovery algorithm constructs exactly one transition per event and introduces no silent transitions. Thus, by construction, it does not create transitions with labels that already appear in the Petri net. Therefore, we exclude the measures C_{dup} and C^{τ}_{dup} from the rest of this analysis, since no factor can influence the values returned by these measures.

Table 4 also shows that both Bartlett's test of sphericity and the test evaluating MSA values succeed. The p-value of the former test is well below 0.05, while all values of the latter test are above 0.6. In turn, the variables feature sufficient intercorrelation to execute a statistically relevant factor analysis. The results of this factor analysis are shown in Table 5. We underline the highest factor loading of a variable if the difference of its absolute value to the absolute values of all other factor loadings for this variable is at least 0.15. Table 5 also shows that all communalities are above the value 0.3, as required. Furthermore, the only variable with a communality below 0.6 is the measure C_{MM}. Thus, from this factor analysis we can derive the dimensions reported in Table 6 with statistical significance.

Compared to the dimensions found by Lieben et al. [7] and reported in Table 3, we observe that the measures C_{size}, C_{MM}, C_{ts} and C_{CFC} again appear together in one dimension. However, when using the Alpha Miner to discover Petri nets, the measures C_{mcd}, C_{CNC} and $C_{\emptyset}$ also belong to this dimension, so C_{mcd} and C_{CNC} are again influenced by the same factor. However, the measure C_{acd} is not influenced by the same factor as C_{mcd} and C_{CNC}. Instead, it belongs to the same dimension as the measure C_{dens}. Similarly, C_{sep} and C_{seq} now belong to the same dimension whereas they appear in separate dimensions in Table 3. C_{CH} is the only measure whose highest factor loading lies in the fourth dimension. However, it is also significantly influenced by the second factor and thus related to the measures C_{acd} and C_{dens}.

Table 4
Descriptive statistics and the results of Bartlett's test of sphericity and the MSA test for the complexity measures when using the Alpha Miner [16] to discover Petri nets.

	Mean	Median	Std	Min	Max
Size, C_{size}	63.44	65	30.16	17	157
Connector Mismatch, C_{MM}	12.49	8	12.73	0	66
Connector Heterogeneity, C_{CH}	0.92	0.94	0.11	0	1
Token split, C_{ts}	38.61	33	38.00	0	193
Control flow complexity, C_{CFC}	52.34	49	40.72	3	228
Separability, C_{sep}	0.93	0.97	0.12	0.24	1.00
Average connector degree, C_{acd}	4.56	4.55	1.03	2.83	11.50
Maximum connector degree, C_{mcd}	10.96	11	4.76	3	26
Sequentiality, C_{sep}	0.95	0.99	0.11	0.38	1.00
Coefficient of network connectivity, C_{CNC}	1.71	1.71	0.45	0.94	3.01
Density, C_{dens}	0.07	0.06	0.04	0.04	0.38
Number of duplicate tasks, excluding τ, C_{dup}	0	0	0	0	0
Number of duplicate tasks, including τ, C^{τ}_{dup}	0	0	0	0	0
Number of empty sequence flows, $C_{\emptyset}$	28.56	27	25.01	0	113

BToS p-value	BToS χ^2-value	Global MSA value
0.0	5 582.07	0.83

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	$C_{\emptyset}$
MSA value	0.83	0.94	0.61	0.90	0.88	0.74	0.62	0.87	0.77	0.92	0.63	0.88

Table 5
The factor loadings of the factor analysis and the communalities of each variable when using the Alpha Miner [16] to discover Petri nets.

	Factor 1	Factor 2	Factor 3	Factor 4
Size, C_{size}	0.9419	−0.1856	0.2135	−0.0609
Connector Mismatch, C_{MM}	0.5648	−0.021	0.1702	0.1525
Connector Heterogeneity, C_{CH}	−0.0756	−0.4894	0.0306	0.6223
Token split, C_{ts}	0.9709	0.0231	0.1405	−0.1295
Control flow complexity, C_{CFC}	0.9682	0.0149	0.1949	−0.032
Separability, C_{sep}	0.2627	−0.0503	0.9279	−0.0967
Average connector degree, C_{acd}	0.278	0.9024	0.1998	−0.1098
Maximum connector degree, C_{mcd}	0.7951	0.4146	0.2369	0.0408
Sequentiality, C_{seq}	0.3538	0.2991	0.7586	0.2011
Coefficient of network connectivity, C_{CNC}	0.921	0.1027	0.3048	0.0255
Density, C_{dens}	−0.4282	0.8121	−0.0496	−0.1656
Number of empty sequence flows, $C_{\emptyset}$	0.9718	−0.0117	0.1912	−0.0869

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	$C_{\emptyset}$
Communality	0.97	0.37	0.63	0.98	0.98	0.94	0.94	0.86	0.83	0.95	0.87	0.99

Table 6
The resulting dimensions of the factor analysis when using the Alpha Miner [16] to discover Petri nets.

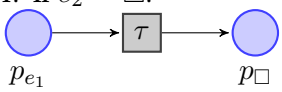
Dimension 1	Dimension 2	Dimension 3	Dimension 4	Excluded
C_{size}, C_{MM}, C_{ts}, C_{CFC}, C_{mcd}, C_{CNC}, $C_{\emptyset}$	C_{acd}, C_{dens}	C_{sep}, C_{seq}	C_{CH}	C_{dup} C^{τ}_{dup}

5.2. Directly Follows Miner

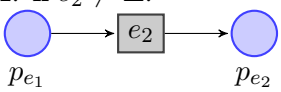
The Directly Follows Miner [17] is a fairly simple discovery algorithm. It starts by constructing a graph whose set of nodes is the set of events in the process, together with one designated start node $\triangleright$ and one designated end node $\square$. Then, it introduces edges between the nodes representing two events, e_1 and e_2, if the event e_2 appears directly after e_1 somewhere in the data. Furthermore, it introduces an edge from $\triangleright$ to an event e if this event can appear at the start of the process and an edge from e to $\square$ if the event can appear at the end of the process. After constructing this so-called *directly follows graph*, it transforms it into a Petri net by performing the following steps:

- Create a place p_e for every node e in the directly follows graph;
- For all edges (e_1, e_2) in the directly follows graph, add the following construct to the already constructed places of the Petri net:

1. If $e_2 = \square$:



2. If $e_2 \neq \square$:



By this definition, the Directly Follows Miner cannot construct transitions with more than one place in its pre- or postset. In turn, a model discovered by the Directly Follows Miner cannot contain any and-splits or and-joins. This property can also be observed in the descriptive statistics in Table 7.

Recall that the measure C_{CH} calculates the entropy of connector types. However, a model N discovered by the Directly Follows Miner cannot contain any and-connectors, so $C_{\text{CH}}(N) = 0$. Similarly, since the net features no parallelism, we do not need to delete any edges to remove all parallelism. In turn, $C_{\text{ts}}(N) = 0$ for any model N discovered by the Directly Follows Miner. Similarly, since there are no and-splits or and-joins in a model N discovered by the Directly Follows Miner, $C_{\emptyset}(N) = 0$.

Since these measures always return constant values for the models in the population, we exclude them from the analysis. Furthermore, we exclude the measures C_{MM} and C_{seq}, as their variable-specific MSA values fall below the threshold 0.6. Thus, these variables are not sufficiently intercorrelated.

Table 7
Descriptive statistics and the results of Bartlett's test of spericity and the MSA test for the complexity measures when using the Directly Follows Miner [17] to discover Petri nets.

	Mean	Median	Std	Min	Max
Size, C_{size}	270.94	263	132.75	69	662
Connector Mismatch, C_{MM}	0.92	0	1.25	0	5
Connector Heterogeneity, C_{CH}	0	0	0	0	0
Token split, C_{ts}	0	0	0	0	0
Control flow complexity, C_{CFC}	240.91	229	128.06	48	623
Separability, C_{sep}	0.99	0.99	0.01	0.93	1.00
Average connector degree, C_{acd}	16.88	16.10	5.61	5.88	32.84
Maximum connector degree, C_{mcd}	31.91	31	11.39	10	59
Sequentiality, C_{seq}	0.99	1.00	0.02	0.86	1.00
Coefficient of network connectivity, C_{CNC}	1.76	1.77	0.09	1.48	1.89
Density, C_{dens}	0.04	0.03	0.01	0.03	0.06
Number of duplicate tasks, excluding τ, C_{dup}	214.06	198	121.58	32	585
Number of duplicate tasks, including τ, C^{τ}_{dup}	215.78	202	121.86	34	587
Number of empty sequence flows, $C_{\emptyset}$	0	0	0	0	0

BToS p-value	BToS χ^2-value	Global MSA value
0.0	10 965.18	0.81

	C_{size}	C_{MM}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	C_{dup}	C^{τ}_{dup}
MSA value	0.87	0.58	0.75	0.86	0.75	0.97	0.55	0.71	0.76	0.91	0.80

Table 8
The factor loadings of the factor analysis and the communalities of each variable when using the Directly Follows Miner [17] to discover Petri nets.

	Factor 1	Factor 2	Factor 3	Factor 4
Size C_{size}	0.8217	0.4347	0.2549	0.2634
Control flow complexity C_{CFC}	0.8317	0.3997	0.2702	0.2718
Separability C_{sep}	0.1946	0.1082	0.1246	0.7523
Average connector degree C_{acd}	0.8342	0.2107	0.4411	0.2204
Maximum connector degree C_{mcd}	0.6099	0.422	0.4801	0.1012
Coefficient of network connectivity C_{CNC}	0.5921	0.1578	0.6874	0.3521
Density C_{dens}	−0.3913	−0.8951	−0.1374	−0.1478
Number of duplicate tasks, excluding τ, C_{dup}	0.8451	0.3778	0.2694	0.2634
Number of duplicate tasks, including τ, C^{τ}_{dup}	0.8429	0.3788	0.2701	0.268

	C_{size}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{CNC}	C_{dens}	C_{dup}	C^{τ}_{dup}
Communality	1.00	1.00	0.54	0.98	0.79	1.00	0.99	1.00	1.00

Table 8 shows the results of the factor analysis when using the Directly Follows Miner to discover Petri nets. All communalities are above the threshold 0.3. Furthermore, most variables have very high communalities, indicating that the discovered factors perfectly describe these variables' variances. However, the measures C_{mcd} and C_{CNC} have more than one significant factor loading, i.e., these variables have *cross-loadings*. Hair et al. [13, p. 119] propose to ignore such variables in the interpretation of the results. However, Lieben et al. [7] choose to assign cross-loaded variables to all dimensions with sigificant loadings. To ensure comparability with their results, we do the same, but color the names of the affected measures in gray so it is easier to identify and ignore them if needed. The dimensions resulting from this factor analysis are reported in Table 9.

Table 9
The dimensions of the factor analysis when using the Directly Follows Miner [17] to discover Petri nets.

Dimension 1	Dimension 2	Dimension 3	Dimension 4	Excluded
C_{size}, C_{CFC}, C_{acd}, C_{dup}, C^{τ}_{dup} C_{mcd}, C_{CNC}	C_{dens}	C_{mcd}, C_{CNC}	C_{sep}	C_{MM}, C_{CH}, C_{ts}, C_{seq}, $C_{\emptyset}$

Like in the results reported in Table 3 and Table 6, the measures C_{size} and C_{CFC} belong to the first dimension. In contrast to the previous results, however, the measures C_{acd}, C_{dup}, and C^{τ}_{dup} are also part of the first dimension. The measures C_{mcd} and C_{CNC} both belong to the first and third dimensions due to their cross-loadings. Considering that these measures also appear together in Table 3 and Table 6, it seems that they are correlated regardless of the used discovery algorithm. The measures C_{dens} and C_{sep} form their own dimensions when we use the Directly Follows Miner to discover Petri nets.

5.3. Inductive Miner

The Inductive Miner [18] is one of the most frequently used discovery algorithms. It returns block-structured Petri nets, which are easy to read, and guarantees that the input language is a subset of the discovered net's language. Similar to the Directly Follows Miner, it first constructs the directly follows graph for the input data. Afterward, it scans this graph for structures that represent parallelism, choices, sequences, or iterations. It transforms these structures, maintaining their order, into a so-called *process tree* that features an intuitive translation to Petri nets. One property of Petri nets that were translated from process trees is that it contains no implicit places. In turn, a Petri net N discovered with the Inductive Miner has $C_{\emptyset}(N) = 0$. Furthermore, each transition label occurs once in such a net N except

Table 10
Descriptive statistics and the results of Bartlett's test of sphericity and the MSA test for the complexity measures when using the Inductive Miner [18] to discover Petri nets.

	Mean	Median	Std	Min	Max
Size, C_{size}	71.68	56	36.64	34	197
Connector Mismatch, C_{MM}	0.30	0	0.77	0	6
Connector Heterogeneity, C_{CH}	0.47	0.58	0.29	0.00	0.88
Token split, C_{ts}	4.10	2	4.57	0	18
Control flow complexity, C_{CFC}	42.50	40	14.78	9	89
Separability, C_{sep}	0.97	0.98	0.03	0.90	1.00
Average connector degree, C_{acd}	7.33	5.28	5.12	2.86	20.00
Maximum connector degree, C_{mcd}	16.97	18	7.83	3	34
Sequentiality, C_{seq}	0.98	1.00	0.06	0.71	1.00
Coefficient of network connectivity, C_{CNC}	1.50	1.50	0.20	1.11	1.82
Density, C_{dens}	0.09	0.06	0.08	0.01	0.25
Number of duplicate tasks, excluding τ, C_{dup}	0	0	0	0	0
Number of duplicate tasks, including τ, C^{τ}_{dup}	20.52	11	20.10	0	81
Number of empty sequence flows, $C_{\emptyset}$	0	0	0	0	0

BToS p-value	BToS χ^2-value	Global MSA value
0.0	6 822.44	0.75

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	C_{dup}
MSA value	0.69	0.76	0.70	0.83	0.57	0.62	0.77	0.67	0.49	0.80	0.75	0.74

Table 11
The factor loadings of the factor analysis and the communalities of each variable when using the Inductive Miner [18] to discover Petri nets.

	Factor 1	Factor 2	Factor 3	Factor 4
Size, C_{size}	0.9712	−0.1515	−0.1211	0.1158
Connector Mismatch, C_{MM}	0.2446	−0.2312	−0.2256	0.2028
Connector Heterogeneity, C_{CH}	−0.0123	−0.1617	−0.9338	−0.206
Token split, C_{ts}	0.9516	−0.2	−0.074	0.0142
Separability, C_{sep}	0.2051	0.152	0.2582	0.6252
Average connector degree, C_{acd}	−0.4835	0.4608	0.583	0.4128
Maximum connector degree, C_{mcd}	−0.1844	0.9532	0.1908	0.1303
Coefficient of network connectivity, C_{CNC}	−0.4728	0.6628	0.2911	0.4778
Density, C_{dens}	−0.5808	0.3288	0.6126	0.401
Number of duplicate tasks, including τ, C^{τ}_{dup}	0.9721	−0.1978	−0.0557	0.0831

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{sep}	C_{acd}	C_{mcd}	C_{CNC}	C_{dens}	C^{τ}_{dup}
Communality	0.99	0.21	0.94	0.95	0.52	0.96	1.00	0.98	0.98	0.99

for τ, which can appear more than once. This means $C_{\text{dup}}(N) = 0$ for every model N discovered by the Inductive Miner. Thus, all data for these two variables is constant, and we cannot analyze them in a factor analysis. We therefore omit $C_{\emptyset}$ and C_{dup} in the following analysis.

As shown in Table 10, Bartlett's test of sphericity succeeds on the collected complexity data. However, the variable-specific MSA values for C_{CFC} and C_{seq} fall below 0.6, so we omit them in the analysis. Recalculating the MSA for the remaining variables results in all values being above 0.6. Thus, the remaining variables are sufficiently intercorrelated, and we can perform a factor analysis. The resulting factor loadings and communalities of each variable are shown in Table 11.

As is evident in the table, C_{MM} has no significant factor loadings, that is, loadings with an absolute value above 0.35. The reason is visible in the communality of C_{MM}. Only 21% of its variance is explained by the factors, so none of the factors are fit to predict the values of C_{MM}. In turn, we exclude C_{MM} from the interpretation of the results. Furthermore, Table 11 shows that C_{acd} and C_{dens} feature cross-loadings. We underline all loadings whose absolute values have a difference of at most 0.15 to the absolute value of the highest loading and take them as significant. This way, we assign C_{acd} and C_{dens} to the first and third complexity dimensions in this analysis. As before, we color the cross-loaded variables in the reported dimensions shown in Table 12 in gray.

Table 12
The dimensions of the factor analysis when using the Inductive Miner [18] to discover Petri nets.

Dimension 1	Dimension 2	Dimension 3	Dimension 4	Excluded
C_{size}, C_{ts}, C^{τ}_{dup}, C_{acd}, C_{dens}	C_{mcd}, C_{CNC}	C_{CH}, C_{acd}, C_{dens}	C_{sep}	C_{MM}, C_{CFC}, C_{seq}, $C_{\emptyset}$, C_{dup}

As with the previous results, the measures C_{size}, C_{ts}, and C_{dup} appear together in one dimension. Similarly, C_{mcd} and C_{CNC} together form the second dimension of complexity. C_{CH} and C_{sep} define their own dimension, like they did in the previous analyses. This might be a hint that these two measures evaluate a different aspect of complexity than the other measures do. The fact that C_{CH} is the only entropic measure and C_{sep} the only one based on graph-theory supports this interpretation.

5.4. Heuristics Miner

As the name suggests, the Heuristics Miner [19] uses heuristics to discover Petri nets. As such, it investigates how often events follow one another and derives parallelism, choices, and sequences from this information. By construction, no Petri net N discovered by the Heuristics Miner contains empty sequence flows, i.e., $C_{\emptyset}(N) = 0$ and every transition label except τ occurs exactly once, so $C_{\text{dup}}(N) = 0$. However, inspecting the MSA values for the remaining measures, shown in Table 13, reveals that the intercorrelation of most variables is insufficient for a factor analysis.

Table 13
The MSA values of the complexity measures if the heuristics miner is used to discover Petri nets.

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	C^{τ}_{dup}
MSA value	0.59	0.56	0.67	0.65	0.75	0.47	0.55	0.46	0.56	0.54	0.83	0.59

It is debatable whether a factor analysis of the remaining four variables yields value. In turn, we will not describe the results of such a factor analysis, even though Bartlett's test of sphericity succeeds with a p-value of 0.0 and the overall MSA value is 0.72. The interested reader is invited to inspect the factor analysis on the remaining variables provided in the repository of the tool described in Section 7.

5.5. Hybrid ILP Miner

The Hybrid ILP Miner [20] discovers a Petri net from underlying data by first constructing a transition for each event name. Then, it constructs an integer linear program (short: ILP) using region theory to find places in the pre- and postset of these transitions. It then creates two silent transitions that ensure that the initial marking and the marking reached when the process ends contain exactly one token. Thus, each Petri net discovered by the Hybrid ILP Miner has exactly one duplicate task, that is, one of the two τ-transitions. In turn, $C_{\text{dup}}(N) = 0$ and $C^{\tau}_{\text{dup}} = 1$ for any Petri net N discovered by the Hybrid ILP Miner. We therefore exclude the measures C_{dup} and C^{τ}_{dup} from the analysis. Furthermore, we exclude the measures C_{sep} and C_{dens} as well, since their MSA value, shown in Table 14, are below

Table 14
Descriptive statistics and the results of Bartlett's test of sphericity and the MSA test for the complexity measures when using the Hybrid ILP Miner [20] to discover Petri nets.

	Mean	Median	Std	Min	Max
Size, C_{size}	38.94	39	12.79	20	70
Connector Mismatch, C_{MM}	2.06	0	3.84	0	23
Connector Heterogeneity, C_{CH}	0.86	0.98	0.29	0	1
Token split, C_{ts}	13.66	6	24.84	0	207
Control flow complexity, C_{CFC}	30.65	20	37.01	1	269
Separability, C_{sep}	0.75	0.84	0.24	0.19	0.97
Average connector degree, C_{acd}	4.09	3.67	1.50	3.00	14.48
Maximum connector degree, C_{mcd}	6.94	6	4.75	3	36
Sequentiality, C_{seq}	0.82	0.88	0.17	0.27	1.00
Coefficient of network connectivity, C_{CNC}	1.57	1.33	0.78	1.00	6.96
Density, C_{dens}	0.08	0.08	0.02	0.04	0.20
Number of duplicate tasks, excluding τ, C_{dup}	0	0	0	0	0
Number of duplicate tasks, including τ, C^{τ}_{dup}	1	1	0	1	1
Number of empty sequence flows, $C_{\emptyset}$	6.63	5	6.10	0	32

BToS p-value	BToS χ^2-value	Global MSA value
0.0	7 266.09	0.84

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{sep}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	C_{dens}	$C_{\emptyset}$
MSA value	0.79	0.96	0.72	0.83	0.89	0.58	0.83	0.91	0.63	0.81	0.47	0.91

Table 15
The factor loadings of the factor analysis and the communalities of each variable when using the Hybrid ILP Miner to discover Petri nets.

	Factor 1	Factor 2	Factor 3	Factor 4
Size, C_{size}	0.38	0.1597	0.893	0.1508
Connector Mismatch, C_{MM}	0.3955	0.2203	0.3413	0.4503
Connector Heterogeneity, C_{CH}	0.1276	0.6222	−0.0269	0.1148
Token split, C_{ts}	0.9461	0.1684	0.2337	0.1026
Control flow complexity, C_{CFC}	0.8846	0.2295	0.3722	0.125
Average connector degree, C_{acd}	0.9596	0.1623	0.1443	0.156
Maximum connector degree, C_{mcd}	0.8658	0.2668	0.2822	0.208
Sequentiality, C_{seq}	0.2248	0.8245	0.3839	−0.0278
Coefficient of network connectivity, C_{CNC}	0.9373	0.2368	0.2306	0.0966
Number of empty sequence flows, $C_{\emptyset}$	0.6737	0.4934	0.3147	0.2293

	C_{size}	C_{MM}	C_{CH}	C_{ts}	C_{CFC}	C_{acd}	C_{mcd}	C_{seq}	C_{CNC}	$C_{\emptyset}$
Communality	0.99	0.52	0.42	0.99	0.99	0.99	0.94	0.88	1.00	0.85

0.6. Since Bartlett's test of sphericity succeeds and the remaining variables have MSA values above 0.6, we can perform a factor analysis on them that yields statistical relevance.

The results of the factor analysis are shown in Table 15. Again, variables with cross-loadings are highlighted in red. This concerns the complexity measure C_{MM}, which is influenced by factors one and four. All variables have a communality of at least 0.45, so we can interpret the resulting dimensions as sufficiently statistically significant. Table 16 reports the complexity dimensions obtained when we use the Hybrid ILP Miner to discover Petri nets.

Table 16
The dimensions of the factor analysis when using the Hybrid ILP Miner [20] to discover Petri nets.

Dimension 1	Dimension 2	Dimension 3	Dimension 4	Excluded
C_{MM}, C_{ts}, C_{CFC}, C_{acd}, C_{mcd}, C_{CNC}, $C_{\emptyset}$	C_{CH}, C_{seq}	C_{size}	C_{MM}	C_{dens}, C_{sep}, C_{dup}, C^{τ}_{dup}

Most complexity measures are part of the first dimension. Surprisingly, this is not the case for the measure C_{size}, which defines the third complexity dimension. C_{CH} and C_{seq} are both part of the second dimension. In the previous analyses, both C_{sep} and C_{CH} formed their own dimension, which is not the case when we use the Hybrid ILP Miner. Furthermore, this is the first time that the measures C_{sep} and C_{dens} did not feature enough intercorrelation with other measures and were therefore excluded.

It would be interesting to perform the factor analysis for the Hybrid ILP Miner on different data sets or with a different number of factors. However, this is out of scope for this paper, which aims to compare the resulting dimensions to those found by Lieben et al. [7]. We will thus discuss the results in the next section and provide a tool for further analyses that is described in Section 7.

6. Discussion

This section discusses the results and the limitations of the factor analyses performed in this paper. Since we closely adhere to the approach proposed by Lieben et al. [7], we inherit its limitations. As such, the authors set the threshold for the variables' communalities to 0.3, so at least 30% of a variable's variance must be explained by the analysis. This allows more complexity measures to be analyzed and interpreted, but the interpretations might not be of statistical relevance. Hair et al. [13, p. 119] suggest a threshold of at least 0.5 for the variables' communalities, i.e., half of a variable's variance must be explained by the analysis. Furthermore, according to Hair et al. [13, p. 119], cross-loaded variables should be excluded from the interpretation of the results. However, following the suggestion of Lieben et al. [7], we assign cross-loaded variables to multiple dimensions of complexity.

To address these limitations, Table 17 shows the results of the previous analyses where variables with communalities below 0.5 and variables with cross-loadings across multiple dimensions are excluded. This table shows that the factors influencing the complexity of process models depend on the chosen method for process discovery. However, we can find some similarities between the results.

Across all the performed analyses, the measures C_{ts} and C_{CFC} always belong to the same dimension, provided their MSA values are sufficiently high. This makes sense considering that both C_{ts} and C_{CFC} punish models containing parallel splits. For discovery algorithms different from the Hybrid ILP Miner, the measure C_{size} also belongs to the same dimension as C_{ts} and C_{CFC}. The rationale behind this is that the probability for a model to have parallel splits increases if the model contains more nodes.

Table 17
The results of the factor analyses per discovery algorithm. Variables with communalities below 0.5 or with cross-loadings are excluded. Note that some dimensions were renamed for easier comparability.

	Dimension 1	Dimension 2	Dimension 3	Dimension 4
Alpha Miner	C_{size}, C_{MM}, C_{ts}, C_{CFC}, C_{mcd}, C_{CNC}, $C_{\emptyset}$	C_{acd}, C_{dens}	C_{CH}	C_{sep}, C_{seq}
Directly Follows Miner	C_{size}, C_{CFC}, C_{acd} C_{dup}, C^{τ}_{dup}	C_{dens}		C_{sep}
Inductive Miner	C_{size}, C_{ts}, C^{τ}_{dup}	C_{mcd}, C_{CNC}	C_{CH}	C_{sep}
Hybrid ILP Miner	C_{ts}, C_{CFC}, C_{acd}, C_{mcd}, C_{CNC}, $C_{\emptyset}$	C_{size}		C_{seq}

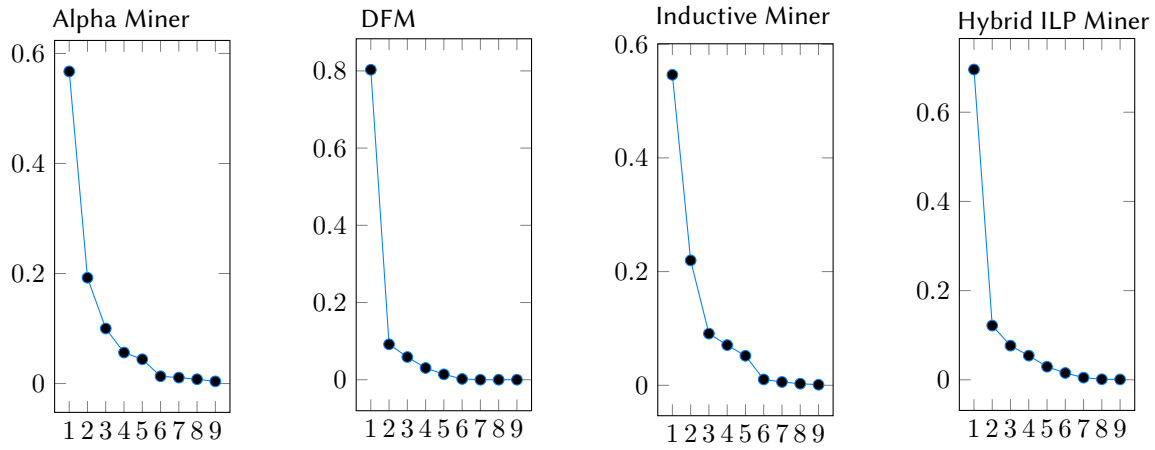


Figure 3: The scree plots for the complexity measures for a fixed process discovery technique.

A second observation is that the measures C_{mcd} and C_{CNC} always appear together in every dimension. For the Directly Follows Miner, both of these measures feature cross-loadings across the same two dimensions, further strengthening this observation. The rationale behind this is, if C_{mcd} increases, there must be a node that gained incoming or outgoing arcs. If this happens without the introduction of new nodes, C_{CNC} must also increase, as it relates the number of arcs to the number of nodes.

We also find that the connector heterogeneity C_{CH} always appears alone in some dimension, provided its MSA value is sufficiently high. This is because C_{CH} is the only entropic measure in the set of investigated complexity measures. In turn, its calculation and interpretation differ from the other measures. Recall that a low value for C_{CH} indicates that the model features one connector type, while a large value means that it features all connector types equally often. In contrast, all other investigated complexity measures return the number or the ratio of complex structures in the process model.

We make similar observations for C_{sep} and C_{seq}. These measures are the only ones based on graph-theoretic properties and hence mostly occur alone in a dimension. The only exception, where both measures appear together in one dimension, is when we use the Alpha Miner for process discovery.

One could argue that another limitation of the performed analyses lies in the number of chosen factors. During our analyses, we closely adhere to the original analysis of Lieben et al. [7] and thus expect 4 factors influencing the investigated complexity measures. However, the appropriate number of factors might also depend on the chosen discovery algorithm. A standard procedure to decide which number of factors is appropriate is the *scree test*. This test calculates the eigenvalues of the factors under the assumption that these eigenvalues would be nearly constant if the data were purely random. The test then sorts the eigenvalues from highest to lowest and displays them in a *scree plot*. The appropriate number of factors is the point on the x-coordinate where the plot starts to "straighten out" [13].

Fig. 3 shows the scree plots for all factor analyses in this paper. According to this criterion, a number of 6 factors would be appropriate. However, setting the number of factors to 6 for the Alpha Miner, the Directly Follows Miner, and the Hybrid ILP Miner leads to no additional insights: All variables' loadings for the factors 5 and 6 are below the threshold 0.35, so none of the investigated measures belong to dimension 5 or 6. The same is true for the Inductive Miner, except for the measure C_{MM}. This measure has its highest loading in the fifth factor. However, as it features a communality of 0.33, we would ignore it in the interpretation of the analysis, leaving dimensions 5 and 6 empty again.

7. Tool Support

We provide the `Python` tool `ComFy` (**Com**plexity Measure **F**actor Anal**y**zer) to reproduce the results of this paper and to further investigate the factors that influence the complexity of Petri nets. The tool and an installation guide are accessible via https://github.com/Pati-nets/ComFy. It allows to choose multiple complexity measures and discovery algorithms to perform factor analyses as shown in this paper.

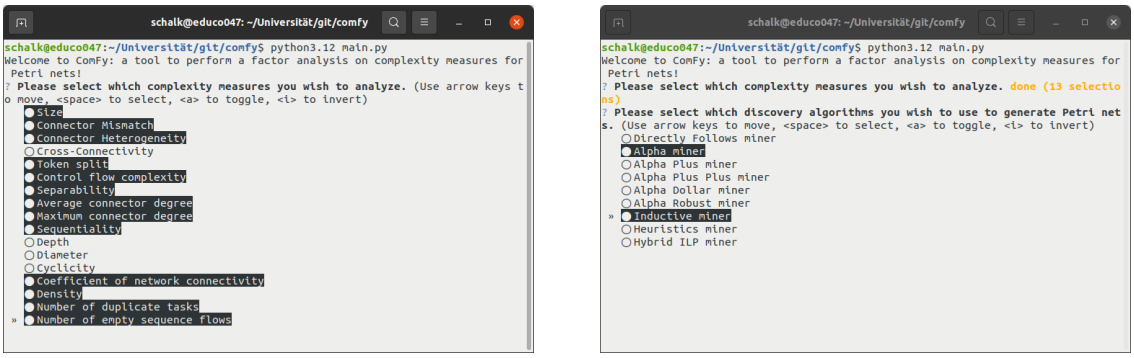


Figure 4: Screenshots showing the options for complexity measures and discovery algorithms in ComFy.

Fig. 4 shows the available measures and discovery algorithms in ComFy. It is possible to select multiple complexity measures and multiple discovery algorithms. After the selection, ComFy will collect complexity data by executing the chosen discovery algorithms for all event logs provided in the folder `input-logs`. It uses the original implementations provided by the ProM framework [21], accessing them via the `Python` library `ProM4Py` [22]. It then calculates all scores of the chosen complexity measures for the discovered models and exports the table of complexity scores into the folder `output`.

Afterward, ComFy starts the factor analysis by first excluding the variables that returned constant complexity scores. It then performs Bartlett's test of sphericity and informs the user whether the test succeeded. If it did not, the user can choose whether to continue the analysis or abort it. ComFy then proceeds by calculating the measure of sampling adequacy for each variable and for the entire set of variables. If the tool finds at any point that the MSA value of a variable is below a specified threshold, it excludes the respective variable and repeats the calculation of MSA values. However, if the overall MSA value is below a given threshold, it aborts the analysis, since it would not be statistically significant.

Figure 5: ComFy's start of a factor analysis.

After these tests, the user can choose to perform a scree test [13, p. 110] to decide on the number of factors for the analysis. Then, the user can choose one out of seven rotation techniques for the factor loadings or to use no rotation at all. Finally, ComFy performs the factor analysis with the selected settings. It offers the option to show the results on the command line using the `Python` library `matplotlib`. Regardless of whether the results are shown on the command line or not, ComFy finally creates a LaTeX-report in the folder `output`. This report contains the descriptive statistics of the generated data, the results of Bartlett's test of sphericity, and all MSA values. Furthermore, it shows the Eigenvalue plot for the scree test and documents the number of factors the user specified. In addition, the report contains the factor loadings found by the analysis in the same way as presented in this paper, as well as a bar diagram for each variable's communality.

Of course, ComFy does not automate the interpretation of the factor analysis, since this is an inherently subjective task. Nonetheless, this tool facilitates the task of performing a factor analysis on complexity measures and enables researchers with and without statistical backgrounds to investigate the factors that influence a Petri net's complexity.

8. Conclusion

This paper performed factor analyses on complexity measures for Petri nets. As a population, it used publicly available event logs to automatically discover Petri nets of different complexity. To investigate the effect of the discovery algorithm on the results of the factor analysis, this paper presented a factor analysis for each of 5 popular discovery algorithms. For each algorithm, it found different factors that influence the complexity of the discovered models. Independently of the discovery technique, the paper found that the size C_{size}, token split C_{ts}, and the control flow complexity C_{CFC} always belong to the same dimension. Thus, it is sufficient to pick one of these measures for a model's complexity evaluation. The same holds for the maximum connector degree C_{mcd} and the coefficient of network connectivity C_{CNC}. The separability C_{sep}, sequentiality C_{seq}, and connector heterogeneity C_{CH}, on the other hand, mostly form their own complexity dimensions.

For future work, we plan to investigate more graph-theoretic measures and measures evaluating the layout of a model. Furthermore, we plan to try other parameters for the factor analysis. The paper facilitates this task by introducing the tool ComFy that enables researchers with and without statistical backgrounds to perform a factor analysis on complexity measures for Petri nets.

Declaration on Generative AI

During the preparation of this work, the author used https://languagetool.org/ to check grammar and spelling. After using this service, the author reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] C. Girault, R. Valk, Petri nets for systems engineering - a guide to modeling, verification, and applications, Springer, 2003. URL: http://www.springer.com/computer/swe/book/978-3-540-41217-5.

[2] I. Koch, W. Reisig, F. Schreiber (Eds.), Modeling in Systems Biology, The Petri Net Approach, volume 16 of *Computational Biology*, Springer, 2011. URL: https://doi.org/10.1007/978-1-84996-474-6. doi:10.1007/978-1-84996-474-6.

[3] W. M. P. van der Aalst, Process Mining - Data Science in Action, Second Edition, Springer, 2016. URL: https://doi.org/10.1007/978-3-662-49851-4. doi:10.1007/978-3-662-49851-4.

[4] W. Reisig, Understanding Petri Nets - Modeling Techniques, Analysis Methods, Case Studies, Springer, 2013. URL: https://doi.org/10.1007/978-3-642-33278-4. doi:10.1007/978-3-642-33278-4.

[5] J. Mendling, Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness, volume 6 of *Lecture Notes in Business Information Processing*, Springer, 2008. URL: https://doi.org/10.1007/978-3-540-89224-3. doi:10.1007/978-3-540-89224-3.

[6] P. Schalk, A. Burke, R. Lorenz, Navigating complexity: Comparing complexity measures with weyuker's properties, in: 6th International Conference on Process Mining, ICPM 2024, Kgs. Lyngby, Denmark, October 14-18, 2024, IEEE, 2024, pp. 145–152. URL: https://doi.org/10.1109/ICPM63005.2024.10680655. doi:10.1109/ICPM63005.2024.10680655.

[7] J. Lieben, T. Jouck, B. Depaire, M. Jans, An improved way for measuring simplicity during process discovery, in: R. Pergl, E. Babkin, R. Lock, P. Malyzhenkov, V. Merunka (Eds.), Enterprise and Organizational Modeling and Simulation - 14th International Workshop, EOMAS 2018, Held at CAiSE 2018, Tallinn, Estonia, June 11-12, 2018, Selected Papers, Lecture Notes in Business Information Processing, Springer, 2018, pp. 49–62. URL: https://doi.org/10.1007/978-3-030-00787-4_4. doi:10.1007/978-3-030-00787-4_4.

[8] M. L. Rosa, P. Wohed, J. Mendling, A. H. M. ter Hofstede, H. A. Reijers, W. M. P. van der Aalst, Managing process model complexity via abstract syntax modifications, IEEE Trans. Ind. Infor-

matics 7 (2011) 614–629. URL: https://doi.org/10.1109/TII.2011.2166795. doi:10.1109/TII.2011.2166795.

[9] V. Gruhn, R. Laue, Reducing the cognitive complexity of business process models, in: G. Baciu, Y. Wang, Y. Yao, W. Kinsner, K. Chan, L. A. Zadeh (Eds.), Proceedings of the 8th IEEE International Conference on Cognitive Informatics, ICCI 2009, June 15-17, 2009, Hong Kong, China, IEEE Computer Society, 2009, pp. 339–345. URL: https://doi.org/10.1109/COGINF.2009.5250717. doi:10.1109/COGINF.2009.5250717.

[10] I. Vanderfeesten, H. A. Reijers, J. Mendling, W. M. P. van der Aalst, J. Cardoso, On a quest for good process models: The cross-connectivity metric, in: Z. Bellahsene, M. Léonard (Eds.), Advanced Information Systems Engineering, 20th International Conference, CAiSE 2008, Montpellier, France, June 16-20, 2008, Proceedings, Lecture Notes in Computer Science, Springer, 2008, pp. 480–494. URL: https://doi.org/10.1007/978-3-540-69534-9_36. doi:10.1007/978-3-540-69534-9_36.

[11] T. Jouck, B. Depaire, Generating artificial data for empirical analysis of control-flow discovery algorithms - A process tree and log generator, Bus. Inf. Syst. Eng. 61 (2019) 695–712. URL: https://doi.org/10.1007/s12599-018-0541-5. doi:10.1007/S12599-018-0541-5.

[12] K. B. Lassen, W. M. P. van der Aalst, Complexity metrics for workflow nets, Inf. Softw. Technol. 51 (2009) 610–626. URL: https://doi.org/10.1016/j.infsof.2008.08.005. doi:10.1016/J.INFSOF.2008.08.005.

[13] J. Hair, W. Black, B. Babin, Multivariate Data Analysis: A Global Perspective, volume 7 of *Global Edition*, Pearson Education, 2010. URL: https://books.google.de/books?id=SLRPLgAACAAJ.

[14] E. Verbeek, Process discovery contest 2025, version 1, 2025. URL: https://doi.org/10.4121/7212a73a-1eac-4a08-8c01-973dca020822.v1.

[15] N. M. Jeremy Biggs, Factor analyzer – a factor analysis tool written in python, 2024. URL: https://pypi.org/project/factor-analyzer/.

[16] W. M. P. van der Aalst, T. Weijters, L. Maruster, Workflow mining: Discovering process models from event logs, IEEE Trans. Knowl. Data Eng. 16 (2004) 1128–1142. URL: https://doi.org/10.1109/TKDE.2004.47. doi:10.1109/TKDE.2004.47.

[17] S. J. J. Leemans, E. Poppe, M. T. Wynn, Directly follows-based process mining: Exploration & a case study, in: International Conference on Process Mining, ICPM 2019, Aachen, Germany, June 24-26, 2019, IEEE, 2019, pp. 25–32. URL: https://doi.org/10.1109/ICPM.2019.00015. doi:10.1109/ICPM.2019.00015.

[18] S. J. J. Leemans, D. Fahland, W. M. P. van der Aalst, Discovering block-structured process models from event logs containing infrequent behaviour, in: N. Lohmann, M. Song, P. Wohed (Eds.), Business Process Management Workshops - BPM 2013 International Workshops, Beijing, China, August 26, 2013, Revised Papers, Lecture Notes in Business Information Processing, Springer, 2013, pp. 66–78. URL: https://doi.org/10.1007/978-3-319-06257-0_6. doi:10.1007/978-3-319-06257-0_6.

[19] A. Weijters, W. {Aalst, van der}, A. {Alves De Medeiros}, Process mining with the HeuristicsMiner algorithm, BETA publicatie : working papers, Technische Universiteit Eindhoven, 2006.

[20] S. J. van Zelst, B. F. van Dongen, W. M. P. van der Aalst, H. M. W. Verbeek, Discovering workflow nets using integer linear programming, Computing 100 (2018) 529–556. URL: https://doi.org/10.1007/s00607-017-0582-5. doi:10.1007/S00607-017-0582-5.

[21] B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, W. M. P. van der Aalst, The prom framework: A new era in process mining tool support, in: G. Ciardo, P. Darondeau (Eds.), Applications and Theory of Petri Nets 2005, 26th International Conference, ICATPN 2005, Miami, USA, June 20-25, 2005, Proceedings, Lecture Notes in Computer Science, Springer, 2005, pp. 444–454. URL: https://doi.org/10.1007/11494744_25. doi:10.1007/11494744_25.

[22] H. Kämmerling, E. G. Rocha, W. M. P. van der Aalst, Prom4py - A python wrapper for the prom framework, in: J. D. Weerdt, G. Meroni, H. van der Aa, K. Winter (Eds.), Doctoral Consortium and Demo Track 2024 at the International Conference on Process Mining 2024 co-located with the 6th International Conference on Process Mining (ICPM 2024), Copenhagen, Denmark, October 15, 2024, CEUR Workshop Proceedings, CEUR-WS.org, 2024, pp. 1–6. URL: https://ceur-ws.org/Vol-3783/paper_367.pdf.