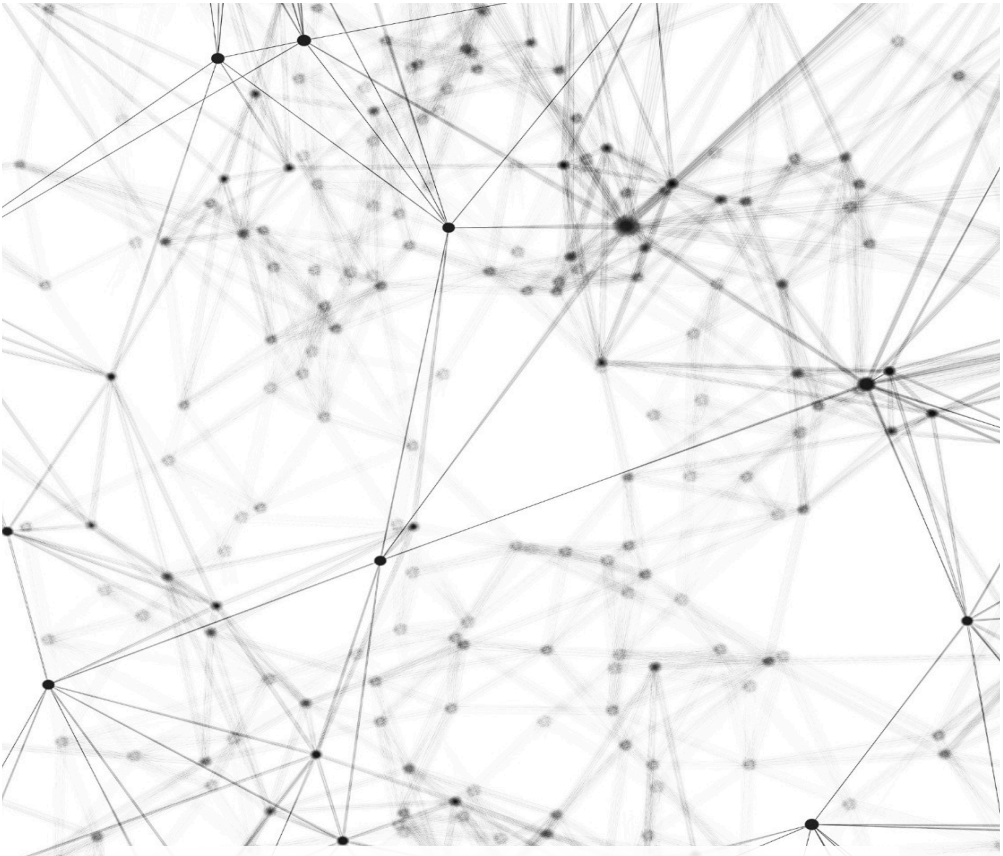
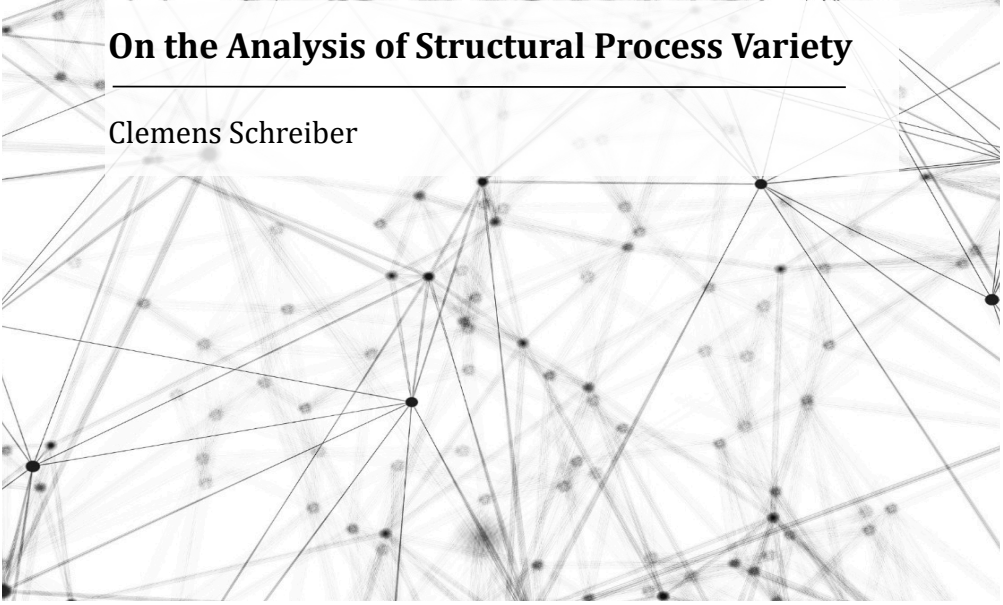


On the Analysis of Structural Process Variety

Clemens Schreiber



Clemens Schreiber

On the Analysis of Structural Process Variety

On the Analysis of Structural Process Variety

by Clemens Schreiber

Karlsruher Institut für Technologie
Institut für Angewandte Informatik und Formale Beschreibungsverfahren

On the Analysis of Structural Process Variety

Zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften (Dr.-Ing.) von der KIT-Fakultät für Wirtschaftswissenschaften des Karlsruher Instituts für Technologie (KIT) genehmigte Dissertation

von Clemens Schreiber M.Sc.

Tag der mündlichen Prüfung: 3. Dezember 2025
Referent: Prof. Dr. Andreas Oberweis
Korreferentin: Prof. Dr. Barbara Weber

Impressum

Autor: Clemens Schreiber

Stand: 5. November 2026



This work is licensed under Creative Commons Attribution-Share Alike 4.0 International License. To view a copy of this license, visit https://creativecommons.org/licenses/by-sa/4.0/

DOI: 10.5445/IR/1000198969

Danksagung

Ich möchte mich herzlich bei Prof. Dr. Andreas Oberweis bedanken, der mich jederzeit unterstützt und bestärkt hat, eigene Forschungsideen zu entwickeln und zu verwirklichen. Ebenso gilt mein herzlicher Dank Prof. Dr. Barbara Weber für ihre kontinuierliche Unterstützung sowie für die Möglichkeit, während meiner Promotion ihre Forschungsgruppe an der Universität St. Gallen besuchen zu können. Darüber hinaus danke ich Prof. Dr. Clemens Puppe und Prof. Dr. Steffen Rebennack für ihre Mitwirkung im Prüfungskollegium meiner mündlichen Promotionsprüfung.

Ein besonderer Dank gilt Amine Abbad-Andaloussi, der mit seiner Erfahrung und seiner großen Leidenschaft für die Forschung eine stetige Quelle der Inspiration für mich war und maßgeblich zum Gelingen dieser Arbeit beigetragen hat. Darüber hinaus möchte ich allen Personen danken, mit denen ich in den vergangenen Jahren das große Glück hatte, zusammenarbeiten zu dürfen. Dazu zählen insbesondere die Kolleginnen und Kollegen der Forschungsgruppe Betriebliche Informationssysteme am AIFB.

Ein großer Dank gilt zudem meiner Familie in Neubrandenburg, Luxemburg und Karlsruhe. Ich danke meinen Eltern, die mich auf meinem Weg zur Promotion stets unterstützt und ermutigt haben. Meiner Familie in Luxemburg danke ich für ihre Herzlichkeit und ihren beständigen Rückhalt. Nicht zuletzt gilt mein Dank Gwendoline und Jonathan, die mein Leben jeden Tag ein wenig schöner und vielfältiger machen.

Abstract

Structural process variety is an inherent characteristic of processes across domains, reflecting the degree of similarity and divergence among alternative execution paths. It directly influences process flexibility, standardization, adaptation, and evolution. In Business Process Management (BPM), understanding structural process variety is essential for evaluating process performance and identifying opportunities for improvement. Beyond BPM, the analysis of structural process variety can also provide valuable insights into human behavior and underlying cognitive processes.

Despite its broad relevance, the development of robust, interpretable, and theoretically grounded measures of structural process variety remains a methodological challenge. This work addresses this challenge by adapting established variety measures from economics and ecology to process analysis. It introduces a measurement framework that quantifies structural process variety based on pairwise distances between process traces, ensuring mathematical rigor and measurement validity.

To further strengthen the assessment of structural process variety, this work also advances the measurement of structural similarity between individual process traces. Existing trace similarity measures often fail to adequately capture structural relationships between traces. To overcome this limitation, novel graph-based similarity measures are proposed that incorporate edge types as a structural abstraction of process behavior. These measures provide a more accurate and structurally

grounded characterization of trace similarity, thereby improving the measurement of process variety.

The proposed measures are evaluated through two empirical studies. The first study demonstrates how structural process variety measures can support business process standardization by identifying trace variants that contribute most to overall variety and by informing decisions on the trade-off between process flexibility and standardization. The second study applies the proposed measures to eye-tracking data and shows that structural process variety and trace similarity provide valuable insights into cognitive effort and information processing. In particular, the proposed similarity measures outperform conventional approaches in detecting transitions between search and inference phases.

Overall, this work presents a theoretically grounded and empirically validated framework for quantifying structural process variety. The framework provides new insights into business process analysis and user behavior analysis.

Contents

Abstract . . . . . **i**

1 Introduction **1**
1.1 Problem Statement 2
1.2 Research Goals 4
1.3 Structure 6

2 Background **9**
2.1 Process Mining 9
2.1.1 Process Structure 11
2.1.2 Structural Process Variety 14
2.2 Measurement Theory 20
2.3 Diversity Analysis 25
2.3.1 General Properties of Diversity 26
2.3.2 Diversity Measurement 31

3 Structural Process Variety **43**
3.1 Related Work 43
3.1.1 Process Flexibility 43
3.1.2 Process Complexity 52
3.1.3 Exploratory Processes 56
3.2 Requirements for Measuring Structural Process Variety . 61
3.3 Measuring Structural Process Variety 65
3.3.1 Structural Variety Measurement Framework . . . 68
3.3.2 Evaluation 78

3.4 Discussion 81
3.5 Summary 83

4 Structural Trace Similarity 85
4.1 Related Work 87
4.1.1 Comparative Trace Analysis 88
4.1.2 Comparative Graph Analysis 90
4.1.3 Process Discovery Algorithms 92
4.2 A Formal Comparison of Similarity Measures 93
4.2.1 Properties Related to Trace Structure 94
4.2.2 Generic Properties 98
4.2.3 Formal Comparison 99
4.3 Graph-based Similarity Measures 103
4.3.1 Enhanced Similarity Measurement with Graph Features 103
4.3.2 Formal Comparison 109
4.4 Empirical Evaluation 110
4.4.1 Evaluation Measures 112
4.4.2 Datasets 113
4.4.3 Sampling Strategy 117
4.4.4 Results 118
4.5 Discussion 120
4.6 Summary 122

5 Structural Process Variety in Business Process Standardization 125
5.1 Related Work 129
5.2 Evaluation 131
5.2.1 Evaluation Method 131
5.2.2 Datasets 133
5.2.3 Results 134

5.3 Discussion 134
5.4 Summary 138

6 Structural Process Variety in User Behavior Analysis . . . 141
6.1 Background on Process Model Comprehension 147
6.1.1 The Effects of Abstraction and Fragmentation in Modularized Process Models 147
6.1.2 Modularization in Process Modeling 149
6.1.3 Comprehension and Cognitive Load 152
6.1.4 Search and Inference Phases 153
6.1.5 Concept Drift Analysis 156
6.2 Related Work 158
6.2.1 Process Model Comprehension from a Task Perspective 158
6.2.2 Information Processing in Process Model Comprehension 160
6.2.3 Concept Drift Analysis to Investigate User Behavior 161
6.3 Research Method 162
6.3.1 Study Design 163
6.3.2 Experiment Procedure 169
6.3.3 Data Analysis 170
6.4 Findings 176
6.4.1 Comprehension and Cognitive Load 176
6.4.2 Search and Inference Behavior 177
6.4.3 Concept Drift Detection 178
6.5 Discussion 179
6.6 Summary 184

7 Conclusion and Outlook 187
7.1 Contributions 187

7.2 Limitations 189
7.3 Outlook 190
7.3.1 Towards a Theory on Software Customization . . 190
7.3.2 Governing Agentic AI Systems 192
7.3.3 Resilience of Information Systems 194

Bibliography **196**

Publication Overview **235**

Nomenclature **237**

List of Figures **242**

List of Tables **250**

A Axioms Related to Weitzman's Diversity Measure 251

B Evaluation Measures for the Comparison of Trace Similarity Measures 253

C Overview of Trace Similarity Measures 255
C.1 Process Flexibility Measures 255
C.2 Process Complexity Measures 260
C.3 Exploratory Process Measures 262

D Experiment Material 265

1 Introduction

Processes can often be executed in multiple different ways, resulting in a variety of alternative execution paths. Structural process variety refers to the structural similarities and differences between these execution paths. In business process management (BPM) and process mining, process analysis aims to identify and visualize such structural differences by generating graphical representations known as process models. These models help illustrate and communicate the extent of structural variety within a process. However, the modeling approach faces several limitations. First, effective interpretation of the models requires a shared understanding of the semantics underlying the graphical notation. Second, high structural variety often results in complex and dense models, which can hinder comprehension and render graphical analysis impractical. Third, objectively comparing the structural characteristics of different process models necessitates a method for quantifying their structural properties.

To address these challenges, this thesis focuses on the quantitative analysis of structural process variety. The introduced measurement approach involves pairwise comparisons of execution paths, resulting in a single numerical value that captures the overall structural variety of a process. This quantitative foundation supports a wide range of analytical techniques, including trace clustering [SGvdA09, DWVBVB13, BCC19], concept drift analysis [BvdAŽP11, PLKH20, AvZRvdA23], and performance analysis [Pen03, MEW10, RBT15].

The following sections outline the key challenges associated with developing quantitative measures for structural process variety and describe how this thesis addresses these issues. The chapter concludes with an overview of the thesis structure.

1.1 Problem Statement

In BPM and process mining, the structure of a process is commonly analyzed based on a respective process model depicting all potential execution paths, or based on an event log depicting a set of executed execution paths. The execution paths are typically represented as traces, i.e., ordered sequences of activities. The structural characteristics of a trace are defined by the ordering and the repetition of its activities. For example, the trace $\langle A, B, C\rangle$ consists of three sequential activities, whereas the trace $\langle A, B, B, C\rangle$ additionally includes a repetition of the activity B, indicating a loop.

Structural process variety encompasses the full range of structural similarities and differences among traces within a given set. For example, the set $L_1 = \{\langle A, B, C\rangle, \langle A, B, B, C\rangle\}$ exhibits greater structural variety than $L_2 = \{\langle A, B, C\rangle, \langle A, B, C\rangle\}$ due to the structural differences between the two traces in L_1.

To actually quantify the structural variety requires evaluating the pairwise structural similarities and differences among all traces in a set. However, existing measures often overlook such structural comparisons, instead focusing on the relative frequencies and the distribution of structural trace properties within the set [Pen03, BDS19, AMVW22]. Moreover, many of these measures lack a theoretical foundation, which hinders the ability to evaluate their validity and to derive meaningful interpretations from their computed results.

Another significant limitation is the lack of trace similarity measures capable of comparing structural properties between traces [BS23]. As illustrated in Fig. 1.1, traces can exhibit diverse structural process patterns [vdATHKB03], including loops (repetitions of activities), skips (omissions of activities), and choices (alternative execution paths). Although some existing similarity measures account for general structural properties, they often fail to capture more nuanced process patterns. This shortcoming reduces both the validity and interpretability of the resulting similarity assessments.

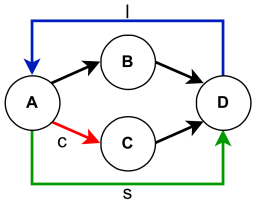


Figure 1.1: Graphical representation of the trace $\sigma = \langle A, B, D, A, C\ D, A, D \rangle$, consisting of the four activities A, B, C, and D. The trace contains three structural patterns indicated by the colored edges: a loop (l), a choice (c), and a skip (s).

A final challenge in measuring structural process variety lies in the fact that the concept itself remains poorly defined and comparatively underexplored, especially when contrasted with more established constructs such as process similarity [DDVD+11, BL12, STF+17] and process complexity [GL06, Men08, AMVW22]. As a result, empirical investigations are essential to validate its practical relevance and to establish its role within the broader context of process analysis.

1.2 Research Goals

To address the challenges outlined in the previous section regarding the measurement of structural process variety, this thesis pursues three primary research goals.

(1) Development of measures for structural process variety. The first objective is to establish valid and theoretically grounded measures for analyzing structural process variety. This requires the identification and formalization of desirable properties for such measures, ensuring their soundness and meaningfulness.

To inform this development, the thesis draws on insights from related disciplines that also investigate the concept of variety. In particular, research in economics and ecology, where the analysis of diversity is a central concern, provides a valuable foundation [Wei92, NP02, Bau06, DBDB18]. In this context, variety is treated analogously to diversity, with the distinction that variety refers to differences within a group, while diversity typically refers to differences between groups. For example, an organization may operate a diverse set of process types (e.g., hiring, accounting, sales), each of which may exhibit internal variety in the form of different execution paths [Pen03].

Despite differences in application, diversity measures across domains often share a common mathematical foundation [Wei92, Bau06, Sti07, Gra08, NP09]. This thesis demonstrates how the mathematical properties of diversity measures, particularly those developed in economics [Wei92, NP02, Gra08, NP09], can inform the design and selection of measures for structural process variety.

(2) Development of measures for structural trace comparison. The second objective focuses on the pairwise comparison of single execution paths, i.e., traces, aiming to define valid similarity measures that capture structural differences and similarities between individual traces.

As with the structural process variety measures, this involves identifying desirable mathematical and structural properties that such measures should satisfy.

A key emphasis is placed on identifying structural features that meaningfully represent the structure of single traces. To extract these features, the thesis leverages graph analysis techniques [CLRS22]. As illustrated in Fig. 1.1, these techniques enable the classification of edges in a directed graph according to specific structural patterns, such as loops, skips, and choices.

Based on this approach, several novel similarity measures are proposed. These measures are designed to capture nuanced structural differences between traces more effectively than existing ones, thereby enhancing the accuracy and interpretability of structural process variety assessments. The proposed measures are subsequently evaluated and compared against alternative measures from the literature.

(3) Validation of structural process variety measures. The third objective is to validate both the conceptual framework and the proposed measures of structural process variety. This validation is conducted in two parts:

- ***Business Process Standardization***: The measures are evaluated for their utility in guiding process standardization efforts. Structural process variety is shown to support decision-making in balancing process efficiency and flexibility, particularly when simplifying a reference process to a limited set of execution paths. The analysis demonstrates that structural variety can serve as a proxy for process flexibility and that the proposed measures can rank traces according to their contribution to overall process flexibility.
- ***User Behavior Analysis***: Given the broad applicability of process analysis to socio-technical systems [vB24], the thesis also explores

the use of structural process variety measures in the context of user behavior processes [DJR+14]. Specifically, eye-tracking data from an experimental study involving comprehension tasks is analyzed. The resulting traces, which represent sequences of fixation points, reveal that structural process variety plays a significant role in understanding user behavior and cognitive processing. The main focus of the variety analysis is to detect concept drifts in user behavior, indicating a transition from information search to information inference.

1.3 Structure

This thesis is structured into seven chapters, progressing from foundational concepts to empirical validation. It begins with theoretical underpinnings in process mining and diversity measurement, followed by the development of structural process variety measures at both the event log and trace levels. The latter chapters present two empirical studies that validate the proposed measures in distinct application domains.

Chapter 2 - *Background*: This chapter introduces the theoretical and methodological foundations of the thesis. It covers essential terminology and notation from process mining, as well as key concepts from measurement theory and diversity theory that inform the development of structural process variety measures.

Chapter 3 - *Structural Process Variety*: This chapter reviews existing literature on structural process variety and its applications in process analysis. It introduces a set of structural variety measures derived from diversity theory, discusses their formal properties, and presents a framework for measuring structural process variety through pairwise trace comparisons.

Chapter 4 - *Structural Trace Similarity*: This chapter defines ten formal properties for evaluating trace similarity measures. Based on observed limitations in existing approaches, a novel graph-based measurement method is proposed. The method is assessed both theoretically and empirically using synthetic and real-world event logs.

Chapter 5 - *Structural Process Variety in Business Process Standardization*: This chapter investigates the relationship between structural process variety and business process standardization. The proposed measures are applied to four real-life event logs representing different business processes to validate this relationship.

Chapter 6 - *Structural Process Variety in User Behavior Analysis*: This chapter demonstrates the applicability of structural process variety in analyzing user behavior through eye-tracking data. It introduces a comprehensive eye-tracking study designed to investigate process model comprehension, presents key findings, as well as the outcomes of a subsequent concept drift analysis. The results of the concept drift analysis highlight the effectiveness of the trace similarity measures introduced in Chapter 4 in detecting changes in user behavior that indicate a transition from information search to information inference.

Chapter 7 - *Conclusion and Outlook*: The final chapter summarizes the main contributions of the thesis and discusses its limitations. It also outlines potential directions for future research in the analysis of structural process variety.

2 Background

This section presents crucial terminology and notation employed throughout the thesis. The research presented in this thesis is grounded in three main research domains: process mining, measurement theory, and diversity analysis.

2.1 Process Mining

Process mining comprises a variety of data-driven techniques aimed at analyzing processes by extracting insights from event logs. An event log is a record of events commonly captured by an information system during the execution of a process [vdA16, vdA22]. Alternatively, an event log can also be generated through process simulation [vdA16, vdA18].

It is assumed that each event possesses three attributes: case identifier, activity name, and timestamp. The *case identifier*, henceforth referred to as *case id*, allows identifying the process instance to which an event belongs. For example, in a job application process, each event could be linked to a unique applicant, who is associated with a distinct case id. The *activity name*, henceforth referred to as *activity*, describes the type of operation, action or task to which the event pertains. Finally, the *timestamp* indicates the order in which events are executed within a case. The following formal definitions are adapted from van der Aalst [vdA16, vdA22].

Definition 1 (Event) *Let $\mathcal{E} = \mathcal{C} \times \mathcal{A} \times \mathcal{T}$ be the event universe where $\mathcal{C}$ is the universe of case ids, $\mathcal{A}$ is the universe of activities, and $\mathcal{T}$ is the universe of timestamps. Any event $e \in \mathcal{E}$ can be characterized by a case id $c \in \mathcal{C}$, activity $a \in \mathcal{A}$, and timestamp $t \in \mathcal{T}$.*

To identify the respective attributes associated with an event, the following attribute-value mappings are defined: $\#_{act}(e) = a$, $\#_{case}(e) = c$, and $\#_{time}(e) = t$. It should be noted that each event could be associated with additional attributes, such as resources or performance indicators. However, the three defined attributes are sufficient for the analysis of the *process structure*, also commonly referred to as the control-flow perspective of a process [vdA16] [1]. To analyze the structure of a process, it is necessary to introduce the notion of a trace.

Definition 2 (Trace, Event Log) *Let $\mathcal{A}$ be the universe of activities and $\mathcal{A}^*$ the universe of activity sequences. A trace $\sigma = \langle \#_{act}(e_1), \#_{act}(e_2), \ldots, \#_{act}(e_n) \rangle = \langle a_1, a_2, \ldots, a_n \rangle \in \mathcal{A}^*$ is a sequence of events corresponding to a common case id $c \in \mathcal{C}$, where all events in σ are mapped to an activity $a \in \mathcal{A}$ and the order of events respects time, i.e., if $e_1, e_2 \in \sigma$ and $\#_{time}(e_1) < \#_{time}(e_2)$, then $e_2 \not\prec e_1$. An event log L is a multiset of traces over $\mathcal{A}$.*

A trace represents a single *instance* of a process, i.e., a complete execution involving a start and an end activity. How an instance is defined depends on the goal of the process analysis and can significantly change the outcome of the analysis. When, for example, analyzing the life-cycle of a product, a process instance could be defined from the initial production of the product to its recycling, or it could be simply defined from the initial production to its delivery to a customer [FvHS+22].

[1] The distinction in terminology is introduced to emphasize that process structure is a fundamental characteristic of any process regardless of the domain, whereas control flow is a concept rooted in software engineering, typically used to describe relationships between software components [All70].

L'
$\sigma_1 = \langle A, B, C, D, E \rangle$
$\sigma_2 = \langle A, C, B, D, E \rangle$
$\sigma_3 = \langle A, C, D, B, E \rangle$

Table 2.1: Example of a flat event log consisting of three trace variants.

The length of a trace is denoted by $|\sigma| = n$ and a_k represents the k^{th} activity in a trace σ, with $1 \leq k \leq n$. Traces with identical activity sequences are classified as one *trace variant* [TLRDM21]. Thus, a single trace variant can represent multiple instances of a process. To obtain the set of trace variants contained in an event log, we can apply the function $f_{flat}(L) = L'$, such that $L' = [\sigma_1, \ldots, \sigma_m]$ lists all trace variants contained in L with $|L'| = m$. Table 2.1 provides an example of such a *flat event log*.

2.1.1 Process Structure

Representing a process through a graphical process model is a widely adopted method for visualizing its structure. Various process modeling languages have been developed for this purpose, including Petri nets, Event-driven Process Chains (EPC), and Business Process Model and Notation (BPMN) [vdA16]. To enable structural variety analysis independent of any specific modeling language, we abstract from the syntactic details of these notations and define a process model as a simple process graph [DDGB09, vdA22].

Definition 3 (Process Graph) *A process graph is a directed graph $G = (N, E)$ with the set N of labeled nodes, representing activities, and the set E of edges, representing directly-follow relationships between the activities. Each process graph G contains start and end nodes $n_{start}, n_{end} \in N$ to model the start and end of a process.*

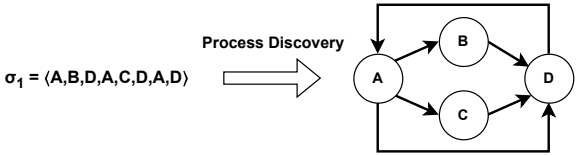


Figure 2.1: Example for the application of process discovery.

This type of process model is also commonly referred to as *directly-follows graph*, and is supported by most commercial and non-commercial process mining tools [vdA22].

A process model may be constructed manually by utilizing the domain knowledge of process stakeholders [VBR14], or it may be automatically derived from event logs through *process discovery* [vdA16, vdA22]. While process discovery is typically applied to event logs containing multiple traces [vdA16], it can also be employed to analyze the structure of individual traces [DGP16]. Figure 2.1 illustrates a process graph discovered from the directly follows relations between activities in the trace σ_1.

The structure of a process is thereby generally determined by the set of activities and their order relation, as reflected in the respective trace variants. Specific sub-sequences of activities within a trace can be defined as *process patterns*. Numerous process patterns have been defined in the literature in order to describe different dependencies between activities in a process, involving different levels of complexity (for an overview, see van der Aalst et al. [vdATHKB03]). For a given trace $\sigma = \langle a_1, a_2, ..., a_n \rangle$ the most basic process patterns can be described as follows [2]:

[2] The description of the patterns follows from the taxonomy of patterns introduced by Bose and van der Aalst [JCBvdA09a].

- Sequence: A sequence is a non-empty subsequence $\lambda = \langle a_i, a_{i+1}, ..., a_k \rangle \subseteq \sigma$ such that all activities in λ are distinct [3]. For example, $\langle A, B \rangle$ shown in Fig. 2.2 (i).
- Loop: A loop is present within a trace σ if there exists a non-empty subsequence $\lambda = \langle a_i, a_{i+1}, ..., a_k \rangle \subseteq \sigma$ such that λ occurs at least twice in σ. For example, $\langle A, B, A \rangle$ shown in Fig. 2.2 (ii). Notably, loops may also overlap, as demonstrated by the trace $\langle A, B, A, B, A \rangle$.
- Choice: A choice indicates that multiple mutually exclusive process execution paths exist within a process. Let $f_{\text{prefix}}(\lambda, k-1) = \langle a_i, a_{i+1}, ..., a_{k-1} \rangle$ be the prefix of a subsequence λ. A choice exist within a trace, if there exist at least two distinct non-empty subsequences $\lambda_1, \lambda_2 \subseteq \sigma$ such that $f_{\text{prefix}}(\lambda_1, k-1) = f_{\text{prefix}}(\lambda_2, k-1)$, but $f_{\text{prefix}}(\lambda_1, k) \neq f_{\text{prefix}}(\lambda_2, k)$. For example, $\langle A, B, D, A, C, D \rangle$ shown in Fig. 2.2 (iii).
- Skip: Let $f_{\text{suffix}}(\lambda, i+1) = \langle a_{i+1}, ..., a_k \rangle$ be the suffix of a subsequence λ. A choice can be additionally considered a skip, if there are at least two distinct non-empty subsequences $\lambda_1, \lambda_2 \subseteq \sigma$ such that $\lambda_1 = \langle a_i, a_k \rangle$ and $\lambda_2 = \langle a_i, ..., a_k \rangle$, $f_{\text{prefix}}(\lambda_1, i) = f_{\text{prefix}}(\lambda_2, i)$ and $f_{\text{suffix}}(\lambda_1, k) = f_{\text{suffix}}(\lambda_2, k)$, with $|\lambda_1| < |\lambda_2|$. For example, $\langle A, B, D, A, D \rangle$ shown in Fig. 2.2 (iv).

It is important to notice that when considering choices and skips within traces, these process patterns always occur in combination with a loop. Fig. 2.2 provides a graphical representation for each of the described process patterns.

[3] For notational convenience, we permit a slight abuse of notation by writing $\lambda \subseteq \sigma$ to indicate that λ is a subsequence of the trace σ.

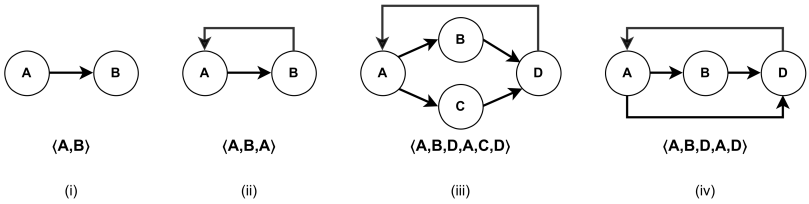


Figure 2.2: Graphical representation of the four structural patterns: (i) sequence, (ii) loop, (iii) choice, and (iv) skip.

2.1.2 Structural Process Variety

In many cases, it can be important to analyze the *structural process variety*, e.g., to evaluate the performance of a process [Pen03, MEW10, RBT15, SAA24], to identify changes in the execution of a process (commonly termed concept drift detection) [BvdAŽP11, PLKH20], to sample event logs [KNGW21], or to discover process models from event logs [MRD+15, BDS19, AMVW22]. In all these cases, the structural variety of a process can be derived based on the structural comparison between the trace variants.

Definition 4 (Structural Process Variety) *The structural variety of a process is defined as a function $V : L \to \mathbb{R}_0^+$, which defines a value for a finite set of traces based on the similarities and differences of structural patterns between all trace in L. It holds that $V(L) = 0$ for all singletons and the empty set.*

For example given the event logs $L_1 = [\langle A, B, C\rangle, \langle A, B, C\rangle]$, $L_2 = [\langle A, B, C\rangle, \langle A, B, B, C\rangle]$, and $L_3 = [\langle A, B, C\rangle, \langle A, B, B, C, D\rangle]$ it can be generally assumed that $V(L_1) < V(L_2) < V(L_3)$, since there exist no structural variety between the traces in L_1, small differences due to the additional loop in the second trace in L_2, and larger differences due to the additional loop and the additional activity in the second trace in L_3.

Whether a process is analyzed based on a single instance (as is the case for the process graph in Fig. 2.1), or multiple instances, depends on the kind of process and the objective of its analysis [vB24]. For example, when analyzing user behavior processes based on eye-tracking [SAAW23, SAAW24, FAAS+24], no two process instances are ever executed in the same way, resulting in a high number of trace variants. However, when analyzing business processes, such as administrative processes [SAA24], instances often follow the same structure, resulting in a lower number of trace variants.

Accordingly, also the objectives of the analysis can change depending on the kind of process. In the context of business process analysis, a common goal is to define a standardized reference process, which ensures a balance between the efficiency and flexibility of a process [MEW10, Tre14, FAM18, SAA24]. In the non-repetitive case, it is often the case that the process is executed in an evolutionary way, independent of a reference process [NSA14]. Here, the analysis will be focused on finding a trade-off between exploration and exploitation, given a defined action space [BT03, GGMH24].

This distinction between the different types of processes is also important for the analysis of structural process variety, since, depending on the context, variety can have a different meaning and can be measured in different ways, as will be discussed in Sect. 2.3. For instance, in a standardized business process the difference in relative frequency of trace variants can be more relevant, than in user-behavior processes, where the structure of the traces constantly changes.

Finally, it should be noticed that the structural process variety can be either derived directly based on an event log, or indirectly based on a process model. If the variety is derived from a process model, one needs to first derive the set of trace variants represented by the model. This can, for example, be achieved based on the simulation of a pro-

cess graph [vdA16, vdA18], thereby identifying all possible execution paths (trace variants) leading from the start node to the end node of the graph. However, one issue with this approach is that the simulation will detect infinitely many trace variants if the graph contains a loop. One way to overcome this issue is to assign probabilities to the execution of activities such that the likelihood of the execution of a loop decreases with each repetition [vdA18] [4].

To measure the structural process variety $V(L)$ for some event log L requires first identifying the similarities and differences between the traces contained in L. This requires the application of a similarity measure.

Definition 5 (Similarity Measure) *A similarity measure $s(\sigma_i, \sigma_j) = (d \circ h)(\sigma_i, \sigma_j)$ is a composition of a distance function d and a feature-extraction function h. Let $\mathcal{F}$ be a universe of features. The function h defines a set of features for each trace $\sigma \in L$, such that $h : L \to \mathcal{P}(\mathcal{F})$, where $\mathcal{P}(F)$ denotes the power set over $\mathcal{F}$. The function d defines a pairwise distance for all traces $\sigma_i, \sigma_j \in L$ according to the derived features, i.e., $d : h(L) \times h(L) \to \mathbb{R}_0^+$.*

A similarity measure s is considered to be a metric, if for all $\sigma_i, \sigma_j, \sigma_k \in L$ the following properties hold:
Property P1. $s(\sigma_i, \sigma_j) \geq 0$ (non-negativity)
Property P2. $s(\sigma_i, \sigma_j) = d(\sigma_j, \sigma_i)$ (symmetry)
Property P3. $s(\sigma_i, \sigma_j) = 0 \Leftrightarrow \sigma_i = \sigma_j$ (identity of indiscernibles)
Property P4. $s(\sigma_i, \sigma_k) \leq s(\sigma_i, \sigma_j) + s(\sigma_j, \sigma_k)$ (triangle-inequality).

In general, one can distinguish between *syntactic* and *feature-based similarity measures* [SGvdA09, ZRSH20, BS23].

[4] Van der Aalst [vdA18] provides the example of a process model consisting of a single activity A, which is repeated with a probability of 0.5. Hence, trace $\langle A \rangle$ has a probability of 0.5, while trace $\langle A, A, A, A, A \rangle$ has only a probability of 0.03125.

Syntactic Similarity Measures

Syntactic similarity measures are calculated directly using the traces without any transformation. In this case h can be considered as an identity function. The pairwise distance d is calculated based on the number of operations required to convert one trace to the other. We will consider edit distance as one particular type of syntactic similarity measure [Lev66].

Definition 6 (Edit Distance) *The similarity measure $s_{Edit}(\sigma_i, \sigma_j)$ based on the edit distance between two distinct traces σ_i and σ_j is determined by optimizing a given cost function, taking into account the insertions, deletions, and substitutions of activities necessary to convert one trace into the other.*

A common edit distance applied in process analysis to measure structural similarities and differences between traces is the Levenshtein distance [Pen03, BCC19, BvdAŽP11, BDS19, BS23]. The Levenshtein distance [Lev66] assigns a value of one for any single-activity edits (insertions, deletions, or substitutions).

An alternative variant of the *Levenshtein distance* is the *normalized Levenshtein distance*, which is also commonly applied in process analysis [JCBvdA09c, JCBvdA09a, DDVD+11, BL12, TSHvdA18, LvdABP21]. The *normalized Levenshtein distance* additionally divides the derived editing costs by the maximum possible number of required edits, i.e., $max(|\sigma_i|, |\sigma_j|)$. This helps to account for loops that may occur in the traces. When, for example, comparing the two traces $\sigma_1 = \langle A, B, B, C \rangle$ and $\sigma_1 = \langle A, B, B, B, B, C \rangle$ the normalized Levneshtein distance provides a more accurate account of the traces' structural similarity.

A similar approach for calculating the distance between two traces is *optimal trace alignment* [JCBvdA10, vdA16, CvDW22, BS23]. Optimal trace alignment is also calculated based on the editing operations nec-

essary to transform one trace into another. However, in contrast to the Levenshtein distance, optimal trace alignment considers matches, mismatches, and gaps between activities. The scores for the matches, mismatches, and gaps can be adjusted according to the context.

For the calculation of the optimal alignments between two traces, Bose and van der Aalst [JCBvdA10] propose the application of the commonly used Needleman-Wunsch algorithm [NW70].

Feature-based Similarity Measures

Compared to syntactic similarity measures, feature-based similarity measures first require a transformation of the traces based on some function $h(\sigma_i) = F_i$ before calculating a distance. A common approach to extract features from a trace is the extraction of *n-grams*.

Definition 7 (n-gram) *Let F_n be a set of sequences of activities with length n. n-gram is a feature-extraction function $h_n : \mathcal{A}^* \to \mathcal{P}(F_n)$ that takes a trace σ and returns a set of all subsequences of length n that exist within σ, i.e., $h_n(\sigma) = \{\langle a_1, a_2, \ldots, a_n\rangle, \langle a_2, a_3, \ldots, a_{n+1}\rangle, \ldots, \langle a_{|\sigma|-n+1}, a_{\sigma-n+2}, \ldots, a_{|\sigma|}\rangle\}$ for $1 \leq n \leq |\sigma|$.*

There exist two types of feature-based similarity measures: set comparison and vector-space embedding. In the case of *set comparison*, the distance between two traces is directly calculated based on the extracted sets of features. One commonly applied similarity measure based on set comparison in process mining is Jaccard distance [SGvdA09, KWW11, KNGW21, BLS22]. Let F_i and F_j be two sets. The Jaccard distance $d_J(F_i, F_j)$ is defined as:

$$d_J(F_i, F_j) = 1 - \frac{|F_i \cap F_j|}{|F_i \cup F_j|} \tag{2.1}$$

where $|F_i \cap F_j|$ is the size of the intersection of the sets F_i and F_j, and $|F_i \cup F_j|$ is the size of their union.

While set comparison is directly calculated based on the extracted features from the traces, *vector-space embedding* additionally requires the mapping of the trace features into a vector space, i.e., $w : \mathcal{F} \to \mathbb{R}^n$. Subsequently, the distance is calculated between the vectors. One similarity measure based on vector-space embedding commonly applied in process analysis is the cosine distance [DDG+15, ZRSH20, BS23]. The cosine distance between two vectors $\mathbf{V}_i$ and $\mathbf{V}_j$ is defined as one minus the cosine similarity. The cosine similarity is given by the dot product of the vectors divided by the product of their magnitudes. Let $\mathbf{V}_i$ and $\mathbf{V}_j$ be two vectors. The cosine distance $d_C(\mathbf{V}_i, \mathbf{V}_j)$ is defined as [HKP12]:

$$d_C(\mathbf{V}_i, \mathbf{V}_j) = 1 - \frac{\mathbf{V}_i \cdot \mathbf{V}_j}{\|\mathbf{V}_i\|\|\mathbf{V}_j\|} \tag{2.2}$$

where $\mathbf{V}_i \cdot \mathbf{V}_j$ is the dot product of the vectors $\mathbf{V}_i$ and $\mathbf{V}_j$, and $\|\mathbf{V}_i\|$ and $\|\mathbf{V}_j\|$ are the magnitudes (or Euclidean norms) of the vectors $\mathbf{V}_i$ and $\mathbf{V}_j$, respectively.

Another commonly applied similarity measure in process mining based on vector-space embedding is the Euclidean distance [JCBvdA09c, SGvdA09, JCBvdA10, ZRSH20, BS23]. The Euclidean distance between two vectors $\mathbf{V}_i$ and $\mathbf{V}_j$ is defined as the square root of the sum of the squared differences between corresponding elements of the vectors. Let $\mathbf{V}_i = (v_{i1}, v_{i2}, \ldots, v_{in})$ and $\mathbf{V}_j = (v_{j1}, v_{j2}, \ldots, v_{jn})$ be two vectors in n-dimensional space. The Euclidean distance $d_E(\mathbf{V}_i, \mathbf{V}_j)$ is defined as [HKP12]:

$$d_E(\mathbf{V}_i, \mathbf{V}_j) = \sqrt{\sum_{k=1}^{n} (v_{ik} - v_{jk})^2} \tag{2.3}$$

where v_{ik} and v_{jk} are the k-th elements of the vectors $\mathbf{V}_i$ and $\mathbf{V}_j$, respectively.

While similarity measures allow evaluating the structural differences between two traces, they do not allow measuring the structural variety of an event log, containing more than two traces. A structural process variety measure must therefore aggregate the structural properties of multiple traces. In order to derive and evaluate such measures, several formal properties are considered in the following sections, that have been defined in measurement theory (cf. Sect. 2.2) and diversity analysis (cf. Sect. 2.3).

2.2 Measurement Theory

Measurement theory is a branch of mathematics that provides a formal framework for assigning numerical values to objects and phenomena in a way that represents meaningful information about them [Rob79, Zus91, Men08]. For example, it can be stated that an event occurred 10 minutes after another event, but it is not meaningful to state that an event occurred 10 percent later than another event. Although this example might appear rather intuitive, when it comes to measuring less well-defined concepts, such as structural process variety, a formal framework can help to avoid ambiguity.

Measurement theory has been extensively applied for the formal analysis of software complexity measures [Zus91]. These measures were later also adapted for the analysis of process model complexity (for an overview, see Mendling [Men08]). This section covers fundamental concepts and definitions of measurement theory as initially presented by Zuse [Zus91], building upon the preliminary work by Robert [Rob79]

First the notion of a *relational system* is introduced. A relational system is a tuple consisting of a non-empty set of objects, as well as a set of

relations and closed binary operations defined on the set. Two complementary relational systems are considered for measurement: one system corresponding to empirical objects, i.e., real-world observations, and one corresponding to formal objects, i.e., numerical values.

Definition 8 (Empirical relational system) *An empirical relational system $R_E = (O, R_1, \ldots, R_n, \circ_1, \ldots, \circ_m)$ consists of a non-empty set of empirical objects O that are to be measured, a set of k-ary empirical relations R_i on O with $i = 1, \ldots, n$, and binary operations $\circ_j$ on O with $j = 1, \ldots, m$.*

Definition 9 (Formal relational system) *A formal relational system $R_F = (Q, S_1, \ldots, S_n, \bullet_1, \ldots, \bullet_m)$ consists of a non-empty set of formal objects Q such as numbers or vectors, k-ary formal relations S_i on Q with $i = 1, \ldots, n$, and closed binary operations $\bullet_j$ on Q with $j = 1, \ldots, m$.*

An example for an empirical and a formal relational system in the context of process analysis is shown in Table 2.2. It is important to notice that the two systems can exist independently of each other. However, a measure is defined as a mapping from empirical objects to formal objects.

Definition 10 (Measure) *A measure μ is a mapping $\mu : O \rightarrow Q$ which yields for every empirical object $o \in O$ a formal object (measurement value) $\mu(o) \in Q$.*

The way in which the relations and binary operations are preserved by the mapping is defined by the scale.

Definition 11 (Scale, Real Scale) *Let $R_E = (O, R_1, \ldots, R_n, \circ_1, \ldots, \circ_m)$ be an empirical relational system, $R_F = (Q, S_1, \ldots, S_n, \bullet_1, \ldots, \bullet_m)$ a formal relational system and μ a measure. The triple (R_E, R_F, μ) is a scale if and only if for all i, j and for all $o_1, ..., o_k, x, y \in O$ the following holds:*

	Empirical Relational System	Formal Relational System
Objects	$\sigma_1, \sigma_2 \in \mathcal{A}^*$	$\mathbb{N}$
Relation	equal or longer than: $\sigma_2 \succeq \sigma_1$	equal or greater than: $\mu(\sigma_2) \geq \mu(\sigma_1)$
Operation	concatenation: $\sigma_1 \circ \sigma_2$	addition: $\mu(\sigma_1 \circ \sigma_2) = \mu(\sigma_1) + \mu(\sigma_2)$

Table 2.2: Example for measuring the length of a trace. The comparison between empirical and formal relational system is based on Zuse [Zus91].

$R_i(o_1, \ldots, o_k) \Leftrightarrow S_i(\mu(o_1), \ldots, \mu(o_k))$ *and* $\mu(x \circ_j y) = \mu(x) \bullet_j \mu(y)$. *When* $Q = \mathbb{R}$*, the set of real numbers, the triple* (R_E, R_F, μ) *is denoted a real scale.*

Following the definition of a scale, a measure μ must preserve all empirical relations when assigning values to empirical objects. Table 2.2 provides an example of such a mapping for measuring the length of traces.

In order to gain insights from calculated measurement values often requires the application of mathematical operations. For example, for the analysis of process structure, it can be of interest whether a trace σ_2 is twice as long as σ_1. Operations on measures that lead to meaningful statements are defined as follows.

Definition 12 (Admissible Transformation) *Let* (E_R, F_R, μ) *be a real scale. A mapping* $g : \mu(E) \rightarrow F$ *is an admissible transformation if and only if* $(R_E, R_F, g \circ \mu)$ *is also a scale.*

Scales can be classified according to their respective admissible transformations. For real scales, Stevens [Ste46] proposes the following classification into five scale types.

- *Nominal scale*: measures classify empirical objects without any quantitative evaluation, e.g., each trace is assigned a unique case ID. Transformations must preserve the uniqueness of the classification.

- *Ordinal scale*: measures provide a ranking of the objects, but the intervals between the ranks are not defined, e.g., ranking traces according to the categories: low variety, medium variety, and high variety. A transformation must preserve the ranking of the traces.
- *Interval scale*: measures provide a ranking of the objects with a meaningful distance between the values. But there is no true zero point, e.g., the timestamp of an event. A transformation must preserve the relative distance between the objects.
- *Ratio scale*: measures include all the properties of the interval scale, but have a true zero point, e.g., the execution duration of a trace. This allows for the calculation of meaningful ratios between measurements. A transformation must preserve the relative distance between the objects.
- *Absolute scale*: measures include all the properties of the interval scale but cannot be subdivided into smaller units, e.g., number of activities within a trace. The only admissible transformation is the identity transformation.

As pointed out by Mendling [Men08] this classification has also been criticized and alternative classifications have been proposed in the literature [VW93]. The criticism focuses on the proposed restriction of the scale types to specific choices of statistical methods, and that the scale types are too narrow to encompass all real-world data, such as counted fractions (e.g., percentages) [VW93]. However, the scale types still provide some information regarding the information content of a measure. It is, for example, clear that an ordinal scale provides less information regarding the structural variety of an event log, than an interval scale, and that for this reason in some cases the latter might be preferable to the first.

While the scale type can provide some orientation for the design of a measure, another issue is its empirical interpretation. Kriz [Kri88] points out that the numerical values derived from a measure have no empirical meaning themselves, unless they are translated into the empirical statements. This process is referred to as interpretation and is an important step in the validation of any measure. Only if the trace of length 20 is also considered longer than the trace of length 10 in the empirical relational system, the measured length values can be considered to have empirical meaning. However, identifying the empirical meaning of measures can be difficult, especially when it comes to the measurement of complex concepts, such as variety. As a solution to this problem, Zuse [Zus91] proposes studying the sensitivity of measures by introducing atomic modifications to the empirical objects. We therefore introduce the definition of an atomic modification in a trace.

Definition 13 (Atomic Modification) *An atomic modification is the change of a trace σ to a trace σ' by a single modification based on adding, deleting, or transferring activities or directly-follow relationships in the trace.*

Definition 14 (Partial property) *The sensitivity of a measure μ to an atomic modification is called a partial property $p \in P$ of the measure. The set P denotes the set of partial properties that completely describe a measure.*

For example, to study the effect of an increase in the length of a trace on a length measure, one could simply add a single event to the trace and compare the results of a measure before and after this atomic modification. When, for example, measuring the length of a trace based on the number of events, the measure will increase by the number of events added to the trace, i.e., one. However, when measuring the length of a trace based on the number of activities, the measure will only increase by one if the added event is associated with an activity that has not already occurred in the trace.

Since it is generally not possible to identify all atomic modifications, it can be sufficient to validate a measure according to an incomplete set of partial properties. This, of course, depends on the particular application domain and context, and the atomic modifications that are relevant to them. The evaluation of structural process variety measures, for example, can be restricted to atomic modifications according to the structural properties defined in the previous section, since these structural properties are commonly considered relevant in the literature. However, the description of a measure based on these atomic modifications would certainly not be complete, as there exist also other structural properties [vdATHKB03].

To gain a better understanding of the empirical meaning and potential interpretations of the concept of variety, we will present some theoretical foundations of the concept in the subsequent section. We then proceed to look at some common variety measurement approaches used in the literature across different research domains.

2.3 Diversity Analysis

Variety can be considered along the same lines as diversity (cf. Sect. 1). There exists a substantial body of research on diversity measures in particular in the field of economics [Wei92, Gra08, NP09] and ecology [Bau06, DBDB18]. Furthermore, the concept of diversity and its measures are applied across many additional research disciplines, such as organizational studies, psychology, physics, and computer science (overviews regarding the concept of diversity across different fields are provided by Stirling [Sti07] and Page [Pag10]). To better understand the concept of diversity, Sect. 2.3.1 reveals some general properties of diversity independently of a specific domain. Sect. 2.3.2 then continues to investigate the perspective of the formal relational system (cf. Sect. 2.2) by introducing generic diversity measurement approaches.

2.3.1 General Properties of Diversity

Diversity is an irreducible property of any system, rather than its individual entities [Sti07]. Whether a system is composed of a single object or many, it can invariably be characterized by its diversity. There are three perspectives that are commonly considered relevant for the analysis of the diversity of a system [NP02, Bau06, Sti07, HK07, Gra08, Pag10, DBDB18]:

- the number of variants
- the relative frequency [5] of these variants
- and the disparity between the variants.

The way a variant is defined depends on the domain and the goal of the analysis. For example, in the case of structural process analysis, we assume that a variant refers to a unique process execution path, i.e., trace variant (cf. Sect. 2.1).

In the example in Fig. 2.3 a variant refers to a means of transportation, i.e., car, bus, or helicopter. Thus, the two sets A and B can be compared according to the three described diversity perspectives. Interestingly, each perspective can result in a different outcome of the comparison. Both sets contain an equal number of variants (two). However, in terms of relative frequency, the diversity is more balanced in set A than in set B (A: $2/4$ cars and $2/4$ buses vs. B: $2/3$ cars and $1/3$ helicopter). Finally, based on the disparity between the variants, it could be argued that set B is more diverse than set A, since a car is more different from a helicopter than from a bus. The example illustrates that the analy-

[5] Relative frequency, especially in the context of biodiversity, can also be described as abundance [Bau06], evenness [PH00, DBDB18], balance [Sti07, LWB19], or entropy [Gra08, Pag10]. In this regard relative frequency (e.g., used by Nehring and Puppe [NP09]) can be considered as a more neutral terminology independent from a specific domain and type of measure.

sis of diversity hinges on the specific diversity perspective and should consequently be informed by theoretical guidance.



Figure 2.3: Two sets of means of transportation, differing in terms of number of variants, relative variant frequencies, and the distinctiveness of the variants. The figure is based on Purvis and Hector [PH00].

In ecology, for example, the relative abundance of species within an ecosystem is recognized as an important indicator of the functioning of an ecosystem [PH00, McC00, Bau06]. On the other hand, in economics it is, for example, the disparity of life-style choices, which is considered relevant for the analysis of a person's well-being [DvH09]. Since the diversity perspective also influences the outcome of a respective analysis, Baumgärtner [Bau06] argues that the selection of a measurement approach requires a prior normative judgment as to what purpose diversity serves in a system. In addition, Baumgärtner proposes to distinguish between a conservative and liberal notion of diversity [6].

The *conservative* view sees diversity as an expression of a system's unity. Here, diversity is considered an inherent characteristic of a system, which might indirectly contribute to the paramount system's properties, such as stability, productivity, or resilience. This view can

[6] According to Baumgärtner [Bau06] these two points of view originate in two philosophical traditions. The conservative view goes back to Gottfried Wilhelm Leibniz (1646-1716) and Immanuel Kant (1724-1804). The liberal view goes back to René Descartes (1596-1650), John Locke (1632-1704) and David Hume (1711-1776).

be linked to the ecological diversity perspective, focusing on relative abundance [Odu53, PH00]. The *liberal* view sees diversity as a value in itself, i.e., a paramount system property, which enables autonomous individuals to freely choose between diverse alternatives. Here, diversity serves the individual subjective preferences of agents, who can select between alternative choices. This view can be linked to the economic diversity perspective, focusing on the disparity of choices [Wei92, NP09]. The disparity is thus more relevant than the relative abundance, since the disparity of choices (or products) can be easily adjusted, e.g., due to the adjustment of production. Furthermore, alternative choices (or products) can be considered as independent of each other, other than that is the case for species in an ecosystem.

The liberal notion of diversity also aligns very well with the notion of structural process variety. For the structural analysis of a process, it is of less interest how frequently a trace variant has been executed, in comparison to how much disparity there is between the trace variants. The disparity provides, for example, insight into the flexibility and performance of a process [Pen03, MEW10, RBT15, SAA24], how a process is changing over time [BvdAŽP11, PLKH20], and how well the structure of a process is represented by the sampling of process data [KNGW21] or the graphical depiction of the process data [MRD+15, BDS19, AMVW22]. In each of these cases, the variety of the trace variants can be rather easily adjusted, due to a selection process of the trace variants, and does not depend on an evolutionary process, such as in the case of species diversity in an ecosystem.

To describe the concept of diversity in more detail, one should consider three main characteristics of diversity in a system: causes, implications, and constraints [Sti07, Pag10].

An important distinction to make, when considering *causes of diversity*, is between systems that are intentionally created and systems that oc-

cur due to an evolutionary process [Pag10]. In the first case, a system is created according to some design decisions, as is the case of a zoo. In the second case, a system evolves over time as a result of incremental adaptations to its environment, as is the case of an ecosystem. At the same time, it is of course possible to identify examples for system diversity with creational as well as evolutionary causes, as would be the case for a wild-life reservoir.

Regarding the *implications of diversity* it is important to point out that diversity can generally have negative and positive system implications. Depending on the connectedness of the entities in a system, commonly recognized positive system implications are:

- exploration of alternatives, eventually leading to some form of innovation or improvement [Urs02, Jen05, Sti07, VdB08, Pag10, KP17]
- providing flexibility and robustness in the face of uncertainty [McC00, Pen03, Sti07, Pag10, RBT15, SAA24],
- accommodating the disparate array of interests and enabling free choice [PX90, DvH09, Sti07],
- sustain diversity [Mor02, Pag10].

However, diversity does not necessarily lead to a better outcome, since a higher number of diverse entities might also require more coordination effort and increase the chance of failure. Depending on the connectedness of the entities in a system, commonly recognized negative system implications are:

- the contribution to complexity, making a system less comprehensible and therefore increasing the probability for human errors [Zus91, Men08, Pag10],
- the increase in inefficiency [Pen03, VdB08, Pag10, RBT15, SAA24],

- an increasing risk for conflicts regarding the distribution of available resources [Pag10, AAGK20].

An additional aspect to consider for the analysis of the diversity of a system is the *constraints of diversity*. Page [Pag10] identifies the following generic constraints:

- the size of the set of the possible (or action space),
- available resources for diversity,
- interdependency between the system's entities and the required coordination efforts,
- selective pressure,
- the ability to function as a system,
- plasticity and rate of adaption.

The differentiation between the causes, implications, and constraints of structural variety provides a useful lens for analyzing its role in process analysis. To demonstrate this, Table 2.3 offers a simplified comparison of two kinds of processes: (1) automated workflows [vdATHKB03, ZWW+10, vdA16], which are executed autonomously by information systems without human intervention, and (2) evolutionary processes [vdAJ00, RW12, NSA14, PLKH20], which evolve over time in response to human-driven changes. In automated workflows, structural variety is primarily the result of deliberate design decisions and is constrained by predefined trace variants. This variety can increase flexibility but may also introduce inefficiencies. In contrast, structural variety in evolutionary processes emerges from internal and external drivers and is constrained by the available action space, i.e., the process structure might only be partially predefined. While it promotes exploration and adaptation, it also incurs exploration-related costs, such as adaptation costs that occur during transition phases [PLKH20]. Overall, the comparison shows that although both domains exhibit structural variety,

	Automated Workflows	Evolutionary Processes
Causes	Design	Internal and external drivers
Constraints	Defined trace variants	Defined action space
Positive Implications	Flexibility	Exploration
Negative Implications	Inefficiency	Exploration costs

Table 2.3: Comparison between the causes, implications, and constraints of structural variety in automated workflows and evolutionary processes.

they differ substantially in how this variety originates, is constrained, and affects system performance.

Of course, real-world examples may deviate from the process categories described herein. For instance, structural variety in a standardized workflow [Tre14, FAM18, SAA24] may arise not only from intentional design choices but also from external factors, such as shifts in the business or technological environment, and internal factors, including design flaws or technical malfunctions [RW12]. A typical example is the introduction of a new information system within an organization, which can trigger unanticipated structural changes [PLKH20].

Despite such deviations, systematically considering the causes, implications, and constraints of structural variety may enhance the understanding of the phenomenon. This, in turn, can support the validity and appropriateness of selecting a particular approach for measuring structural variety.

2.3.2 Diversity Measurement

There exists a dispute over the extent to which a measure should adhere to predefined axioms, since it carries the risk of creating well-designed measures without properly reflecting reality [BEM96, Men08]. However, defining the formal properties of a potential diversity measure can be helpful in particular, if the concept of diversity is not fully

delineated by theory [Gra08]. This section identifies four different approaches for the measurement of diversity and their respective properties: number of variants, relative frequency of variants, disparity between variants, and the aggregation of frequency and disparity. Following Baumgärtner [Bau06], a diversity measure can generally be defined as follows:

Definition 15 (Diversity Measure) *Let $\Omega = \{S, n, F, m, x, p\}$ be a system, where:*

- *S denotes a set of variants, such that $S = \{s_1, s_2, ..., s_n\}$*
- *$n \in \mathbb{N}$ denotes the number of different variants contained in S*
- *F denotes a set of features describing the properties of the variants, such that $F = \{f_1, f_2, ..., f_m\}$*
- *$m \in \mathbb{N}$ denotes the total number of features, which allow to distinguish between variants*
- *Let x_{ij} be the description of variant s_i in terms of features f_j, so that $x = \{x_{ij}\}_{i=1,...,n;j=1,...,m}$ is the complete characterization of all variants in terms of all features*
- *Let $a_i \in \mathbb{N}$ denote the number (absolute frequency) of variant s_i in S, such that $p_i = \frac{a_i}{\sum_{i=1}^{n} a_i}$ denote the proportion (relative frequency) of a variant s_i in S.*

A diversity measure of a system Ω is a real-valued function D defined on Ω, such that $D : \Omega \to \mathbb{R}$.

Number of Variants

A simple way to derive a measurement of diversity is to count the number of variants contained in a system.

$$D_{\text{Number}}(\Omega) = n \tag{2.4}$$

The measure accounts for the number of distinguishable variants within a system Ω. Hence, the variants in S are considered completely equal or completely unequal. The measure is commonly applied in ecology to account for the number of species within an ecosystem [McC00], and in software engineering, as well as BPM, to account for the complexity of a software based on the number of lines of code [Zus91, BEM96], or the complexity of a process model based on its number of elements [CMNR06, Men08].

The measure can be classified as absolute scale, allowing for any arithmetic operation, including addition, subtraction, multiplication, and division, as well as calculating ratios (cf. Sect. 2.2). Furthermore, it allows for the ranking of sets, according to the calculated value [Gra08]. However, the measure does not provide any insight into the relative frequency or disparity of variants. This has been recognized and criticized across different research domains [Wei92, NP02, Sti07]. Nehring and Puppe [NP02] point out that a diversity measure should in general not be additive, which they illustrate based on the following quote: *"An angel is more valuable than a stone. It does not follow, however, that two angels are more valuable than one angel and a stone". — Thomas D'Aquinas, Summa contra Geniles, III*. Thus, emphasizing that the disparity between variants and non-additivity are essential to the concept of diversity [7]. Other research disciplines, particularly ecology, emphasize the relevance of the relative frequency of variants, i.e., the balance between variants, for the evaluation of diversity [PH00, Sti07, Mag13, DBDB18]. Accordingly, the following described measurement approaches offer ways to integrate disparity and relative frequency into the measurement of diversity.

[7] Similarly, Briand et al. [BEM96] argue that in the context of software complexity, the complexity of a program composed of two blocks, does in general not equal the mere sum of the individual complexities of those blocks.

Frequency-based Measures

Frequency-based measures can be subsumed under the generalized function [Hil73]:

$$D_{\text{Entropy}}(\Omega) = (\sum_{i=1}^{n} p_i^{\alpha})^{\frac{1}{1-\alpha}} \quad \text{with} \quad \alpha \geq 0 \tag{2.5}$$

where the parameter α determines how much weight the diversity measure places on the relative frequency of the variants. For $\alpha = 0$, the equation yields (2.4), the number of variants. In this case, the relative frequency does not matter. A common alternative is the Shannon-Wiener entropy [Bau06, Sti07, NP09, Gra08, DBDB18], which can be obtained from (2.5) by setting $\alpha = 1$ (a proof is provided by Hill [Hil73]):

$$D_{\text{Shannon-Wiener}}(\Omega) = e^{H} \quad \text{with} \quad H = -\sum_{i=1}^{n} p_i \ln p_i \tag{2.6}$$

The Shannon-Wiener entropy satisfies the following mathematical properties [Sha48, Sti07, Gra08, NP09, Pag10, DBDB18]:

1. *Monotonically increasing with respect to n under uniform distributions*: the higher the number of variants in a uniform distributed set, the higher the diversity
2. *Continuous with respect to p_i*: the diversity can be evaluated for any distribution of variants, without jumps or breaks in the value
3. *Symmetry*: the calculated value is independent on how the relative abundance is distributed among the variants, e.g. a population of 1 butterfly and 99 birds has the same diversity value as a population of 99 butterflies and 1 bird
4. *Maximum of equality*: exists when variants exist in equal proportions, i.e., $p_i = \frac{1}{n}$ for all i

5. *Scale invariance*: a set, where each variant has 1000 instances, has the same diversity as a set, where each variant has 10 instances
6. *Decomposability*: when a set of variants is divided into disjoint subsets, the total diversity of the original set is equal to the sum of the diversity between the subsets and the diversity within each subset [8].

Although Shannon-Wiener entropy is commonly applied across different research disciplines, particularly in ecology [Bau06, Sti07, Gra08], it is also necessary to consider some of its shortcomings. First, finding a meaningful interpretation of the parameter α can be difficult [NP09], especially since Shannon-Wiener entropy originates from coding theory [Sha48]. In this domain, entropy denotes the minimum average number of bits required to encode without redundancy a randomly drawn member from a population. Hence, it quantifies the uncertainty involved in predicting the value of a random variable. This theoretical reasoning behind the measure must be considered when applied in other domains.

However, this might not always be possible as in the case of the comparison of diversity between sets [DBDB18]. When, for example, comparing a set S_1 of five equally common variants with a set S_2 of ten equally common variants, one could assume that S_2 is twice as diverse as S_1. However, the Shannon-Wiener entropy results in $D_{\text{Shannon-Wiener}}(S_1) = 1.61$ and $D_{\text{Shannon-Wiener}}(S_2) = 2.30$, which does not follow the presumable intuition that S_2 is twice as diverse as S_1 [DBDB18]. Similarly, if S_3 would contain 100 equally common variants, a decrease of 90% of all variants in this set, would result in a decrease of the Shannon-Wiener entropy by around 50% ($D_{\text{Shannon-Wiener}}(S_2) =$

[8] As pointed out by Gravel [Gra08] this property allows to evaluate the diversity of subsets of variants, such as genera or families of species in ecology, independently from each other in order to derive an overall diversity value, which might be considered as a computational advantage.

4.61). Even though in some cases, such as in ecology, a reduction of 90% of all variants (species) might be considered as rather extreme [DBDB18].

In some cases, also *scale invariance* might be deceptive. When, for example, considering the probability of survival of species in an ecosystem, it can make a difference whether each species has 1000 or only 10 individuals [Gra08].

Finally, one can also question the *symmetry* property, as some variants might contribute more to the diversity than others independently of their relative frequency [Gra08]. This can, for example, also be observed in process analysis, where often 80% of the process structure can be explained by 20% of the trace variants or activities, which is also described as Pareto principle [vdA20]. Nevertheless, also infrequent trace variants can contribute significantly to the overall diversity (structural variety) of the process [SAA24].

Thus, to overcome these issues, it is necessary to also consider the similarities and differences between variants for a diversity evaluation. This is possible with disparity-based measures.

Disparity-based Measures

There exist two types of disparity-based measures described in the literature on diversity analysis [Bau06, Sti07, Gra08, NP09, Pag10]: *distance-based measures*, and *attribute-based measures*. The measures differ according to the way they account for the disparity of the variants in a system.

Distance-based measures were initially proposed by Weitzman [Wei92]. They are calculated in two steps:

1. the calculation of pairwise distances between all variants in a set (cf. Sect. 2.1.2)

2. the aggregation of the derived distances to an overall variety value of the set.

One challenge of this approach is to select an appropriate distance function to quantify the similarities and differences between the variants within a set (cf. Sect 2.1.2).

Once, a pairwise distance d is calculated for all variants $s_i, s_j \in S$, the distance between a variant $s_i \in S$ and a subset of variants $Q \subset S$ can be derived as follows:

$$\delta(s_i, Q) = \min_{s_j \in Q} d(s_i, s_j) \text{ for all } s_i \in S \setminus Q. \tag{2.7}$$

This distance indicates the difference in variety between Q and the enlarged set $Q \cup \{s_i\}$. Consequently, the overall structural variety of S can be derived based on (3.1) by iteratively adding up the distances. To achieve this, Weitzman [Wei92] proposes a recursive formula, starting with an arbitrary variant $s_i \in \Omega$ and an arbitrary start value $D_{\text{Weitzman}}(\{s_0\})$:

$$\begin{aligned} D_{\text{Weitzman}}(Q \cup \{s_i\}) = D_{\text{Weitzman}}(\Omega) + \delta(s_i, Q) \\ \text{for all} s_i \in S \setminus Q \text{ and } \varnothing \subset Q \subset S. \end{aligned} \tag{2.8}$$

One important issue with this recursive approach is that, in general, the outcome of (3.2) is not unique, i.e., the diversity value can differ based on the selection of s_0. Weitzman [Wei92] shows that this issue can only be overcome if the applied distance function d is an *ultrametric*, i.e., a metric with the additional property that the two greatest distances among three points are always equal, i.e:

P5: $d(s_i, s_k) \leq max\{d(s_i, s_j), d(s_j, s_k)\}$ *(ultrametric-inequality)*[9].

[9] The equality of the greatest distances between three points follows from P5. This can be shown when considering the three points A, B, C and the distances between them

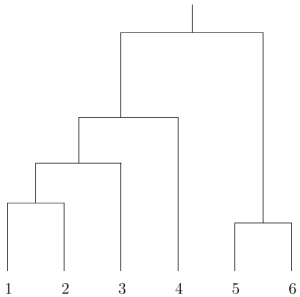
This is a stronger condition than the triangle-inequality (P4) and restricts the evaluation of disparity. Indeed, the work by Nehring and Puppe [NP02, NP09] shows that this restriction generally applies when measuring diversity based on the pairwise comparison of variants.

However, ultrametric distances can be represented by a taxonomic tree, which introduces a hierarchical clustering of the variants. In some domains, such hierarchical clusters might occur naturally due to the characteristics of the variants [Wei92, NP02]. In ecology, for example, species can commonly be categorized according to a taxonomic tree [Wei92]. In such a taxonomic tree all species are categorized according to subgroups of species that each share a particular common ancestor. Thus, representing the evolutionary relationships between species. Fig. 2.4 gives an example of such a taxonomic tree including the ultrametric distances between the species. In the example, originally provided by Weitzman [Wei92], the species sets $\{1, 2\}$, $\{1, 2, 3\}$, $\{1, 2, 3, 4\}$, and $\{5, 6\}$ each form subgroups, which share a common ancestor.

The distance-based measure satisfies the following mathematical properties [Wei92]:

1. *Monotonically increasing with respect to* n: the higher the number of variants in a uniform distributed set, the higher the diversity
2. *Monotonically increasing with respect to* $d(s_i, s_j)$: the diversity can be evaluated for any distribution of variants, without jumps or breaks in the value
3. *Continuous with respect to* $d(s_i, s_j)$: the diversity can be evaluated for any set of variants and their defined distances, without jumps or breaks in the value

$d(A, B)$, $d(B, C)$, $d(A, C)$. It can be either assumed that $d(A, B) = d(B, C) = d(A, C)$, or that one distance is shorter than another. When assuming the second case with $d(A, B) < d(B, C)$ it follows that $d(B, C) = d(A, C)$, since $d(A, B) < d(B, C) \leq \max\{d(A, B), d(A, C)\} = d(A, C)$, and $d(A, C) \leq \max\{d(A, B), d(B, C)\} = d(B, C)$.



(i)

Matrix of ultrametric distances

	1	2	3	4	5	6
1	0	159	250	357	532	532
2	159	0	250	357	532	532
3	259	250	0	357	532	532
4	357	357	357	0	532	532
5	532	532	532	532	0	126
6	532	532	532	532	126	0

(ii)

Figure 2.4: An example of (i) a taxonomic tree describing the evolutionary relations between six different species. The ultrametric distances between the species are depicted by (ii) the distance matrix. In the example, originally provided by Weitzman [Wei92], the taxonomic tree represents the following species: 1 - human, 2 - common chimpanzee, 3 - gorilla, 4 - orangutan, 5 - common gibbon, and 6 - siamang gibbon. The figure is adapted from Weitzman [Wei92].

4. *Link property*: the extraction of some variant s_i from S will reduce the diversity of S by exactly the distance of that removed variant and another remaining variant in S
5. *Twin property*: the diversity does not increase when an already existing variant is added to the set
6. *Maximum diversity that can be added by a variant*: is the maximal distance $d(s_i, s_j)$ between $s_i \notin S$ and some $s_j \in S$.

Some additional properties of the distance-based measure are discussed by Bossert et al. [BPX03] and Gravel [Gra08]. Bossert et al. [BPX03] provide an axiomatic approach to describe the mathematical properties of $D_{\text{Weitzman}}(\Omega)$. The precise description of the identified axioms can be found in the appendix A. Gravel [Gra08] argues that especially the identified axiom called *link indifference* renders the application of $D_{\text{Weitzman}}(\Omega)$ for the measurement of diversity disputable. The axiom *link indifference* (cf. appendix A) can be described as follows

[Gra08]. Assuming that the distance between two species a and c is 6 and the distance between two other species x and y is 10. When now adding a third species b to the set $\{a, c\}$ with the property that the distance between a and b is 6 and the distance between c and b is 4, then the axiom states that the sets $\{x, y\}$ and $\{a, b, c\}$ are of equal diversity. Even though this result might be disputable [Gra08], in the context of the hierarchical structure of a taxonomic tree, it appears reasonable that a third species increases the diversity of a subset, such that it equals the diversity of another subset consisting of only two species.

Attribute-based measures were initially proposed by Nehring and Puppe [NP02] and offer a more general alternative to distance-based measures, as they do not require the definition of a similarity measure. The measurement approach only accounts for the unique features (attributes) represented by a set of objects. This is based on the presumption that the number of distinct features accounts better for the diversity of a set than the number of variants. The predefined features of the system Ω are added up using a weighting function:

$$D_{\text{Nehring-Puppe}}(\Omega) = \sum_{i=1}^{n} \lambda_f \tag{2.9}$$

where λ represents weights over the different features $f \in F$. This generic approach can be easily applied in different disciplines. The flexibility, however, comes at the cost of having to preselect a set of features and weights.

As stated in [Pag10], depending on whether the weights originate from an objective criterion or a subjective criterion, one has either a diversity measure or a value function. The weights should therefore reflect the purpose of the diversity analysis.

In addition to the decision on relevant features and weights, the measurement approach has further shortcomings. Gravel [Gra08] points

out that the distribution of features is not considered by the measure, i.e., it is irrelevant whether the features belong to a single variant or are distributed among multiple variants. This is, for example, an issue in the case of structural variety analysis (cf. Sect. 2.1), since it might be counterintuitive to state that a singleton has a higher diversity than a set of multiple variants.

Two additional critical aspects of the applicability of the multi-attribute approach are the *transferability of features* and the *separability of their functions* [Pag10]. While in software engineering it appears reasonable that different features can be transferred from one software to another [JZWL16], for species in an ecosystem it is not possible to simply adapt new features. Hence, ecologists are more concerned with preserving species than traits. Similarly, it might be the case that features cannot be easily considered separate from each other, as is the case for cross-cutting features in software engineering, such as user authentication [JZWL16]. Consequently, an attribute-based measurement appears only sensible in cases where the features' functionalities are separable.

Finally, it is also possible to define hybrid measures that consider the number, relative frequency and disparity of variants in an aggregated form.

Hybrid measures

Stirling [Sti07] argues that all three diversity dimensions build on each other and should therefore be considered for diversity evaluation. Indeed, it is first necessary to consider the (at least binary) distinctiveness between variants, before one can consider the relative frequency and disparity between them. A hybrid measure, accommodating all three diversity dimensions, was originally proposed by Rao [Rao82] and later derived under somewhat different criteria (independently

from Rao) by Stirling [Sti07]. The measure is defined as a weighted sum of pairwise distances, where the weights are determined by the relative frequencies of the variants. A generalized form can be defined as follows.

$$D_{\text{Rao}}(\Omega) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij})^{\alpha} (p_i p_j)^{\beta} \tag{2.10}$$

where α and β represent weights for the disparity and relative frequency of variants, which can be assigned any values between 0 and 1. Similarly to the frequency-based measures (cf. Sect. 2.3.2) this measure fulfills *symmetry*, *decomposability* and *scale invariance*. Furthermore, the measure exhibits the following properties (based on Stirling [Sti07]):

1. $D_{\text{Rao}}(S) = 0$ if S is a singleton (i.e., the distance is zero)
2. all else being equal, D_{Rao} increases monotonically with the variants' number, relative frequency and disparity
3. symmetric accommodation of diversity perspectives (number, relative frequency and disparity)
4. as simple in structure and as minimal in form as needed to support all three diversity perspectives
5. based on α and β the measure permits to make the diversity perspectives explicit according to the context or the objective of the analysis.

While these properties emphasize the advantages of the Rao-Stirling diversity measure, it has also been criticized that aggregating different diversity perspectives can lead to misleading results [DBDB18, LWB19]. This is due to the fact that the perspectives can lead to contradicting diversity evaluations (as was shown in Sect. 2.3.1 based on Figure 2.3).

3 Structural Process Variety

This chapter further elaborates on the concept of structural process variety. First, Sect. 3.1 identifies how structural process variety is measured and applied in process analysis by examining current research literature in BPM and process mining. Subsequently, in Sect. 3.3.1 a novel framework for the measurement of structural process variety is introduced and evaluated.

3.1 Related Work

In BPM and process mining, structural process variety is generally considered an important process characteristic. Structural variety has been commonly associated with process flexibility [Pen03, Gün09, BvdAŽP11, BDS19, PLKH20, SAA24], process complexity [BWBBG14, AMVW22], and exploratory processes [MRD+15, SAAW23, SAAW24].

3.1.1 Process Flexibility

Flexibility can be considered as the ability of a process to handle anticipated and non-anticipated changes in the operating environment [SMR+08a, RW12]. Changes of a process, which might be anticipated at the design time of a process, can for example be caused by differing country-specific regulations, seasonal changes, or differing customer requirements [RW12]. Changes, which might not be anticipated at the design time of a process, can often be attributed to changes in the business context, legal context, and technological context [vdAJ00,

PLKH20]. From a BPM perspective, one can distinguish between four different process flexibility types [SMR+08a, RW12]. According to Reichert and Weber [RW12] these are [1]:

- *Variability*: Changes can be identified during the design time of a process by analyzing the contextual factors, such as variability in products and services, regulatory requirements, temporal constraints, and customer requirements.
- *Looseness*: Changes cannot be anticipated, since processes in this category are characterized as non-repeatable (i.e., no identical traces within an event log), unpredictable (i.e., the exact course of action cannot be known at the beginning of the process), and emergent (i.e., the exact course of action only emerges during execution, when additional information has been accumulated). One example, of such a process, would be an agile software development process [AKS+21], where the exact order of development steps is not specified in the beginning and each development process is different from one another.
- *Adaptation*: Changes can be caused by special situations during process execution, such as the occurrence of an emergency case in the hospital, or by exceptions, such as the occurrence of multiple emergency cases in the hospital at the same time, e.g., during a pandemic. In contrast to Looseness, some of these special situations and exceptions can be anticipated, and therefore be planned for at the design of a process. However, it is not possible to anticipate all possible exceptions, such that additional support is needed if it is not possible to prepare for such exceptions.

[1] Schonenberg et al. [SMR+08a] make a similar distinction between types of process flexibility according to its causes, i.e., *flexibility by design*, *flexibility by deviation*, *flexibility by underspecification*, and *flexibility by changes*. These flexibility types can be mapped to the ones defined by Reichert and Weber [RW12]. However Reichert and Weber provide a more generic terminology by describing the needs for flexibility within an information system to support less structured business processes.

- *Evolution*: Changes are driven by external factors inherent to the operational environment of a process [vdAJ00, PLKH20]. However, there can also exist internal drivers for changes, such as design errors (e.g., if a process cannot be executed as originally intended), technical problems (e.g., issues related to the information system's infrastructure), or poor internal quality of process models (e.g., ambiguity in a process model [FAAS+24]).

Each of these flexibility types can be associated with a change in process structure, resulting in alternative process execution paths [SMR+08a, RW12]. Hence, high structural process variety can be associated with high flexibility, and low structural process variety with low flexibility. Although Schonenberg et al. [SMR+08a] and Reichert and Weber [RW12] propose different methods on how to support the different flexibility needs based on the design of process models, they do not propose specific measures for the evaluation of flexibility.

Nevertheless, the relevance of flexibility for the performance of a process and the necessity to evaluate flexibility has also been recognized in the literature [Pen03, DRMR13, VLS16]. In BPM four main process performance perspectives are generally considered [DRMR13]: *time-related*, *cost-related*, *quality-related*, and *flexibility-related* performance. Yet, only a few flexibility measures have been proposed in the literature.

In a comprehensive literature review, van Looy and Shafagatova [VLS16] identify 140 process-related performance measures covering all four performance perspectives. But they identify only one measure related to flexibility. Furthermore, the identified measure simply involves the qualitative evaluation of whether an organization has the capability to manage (non-technical) business process anomalies [LM05].

Nevertheless, some more elaborate flexibility measures have been proposed by Pentland [Pen03], Günther [Gün09], Back et al. [BDS19], Bose

Flexibility

Type	Name	Reference
Number	Lempel-Ziv complexity	[Pen03]
	Number of events	[Gün09]
	Number of activities	[Gün09]
	Number of traces	[Gün09]
	Activities per trace	[Gün09]
	Completness of directly follows relations	[Gün09]
	Number of acyclic traces	[PLKH20]
Frequency	Deviation from random	[Pen03]
	Trace entropy	[BDS19]
	Prefix entropy	[BDS19]
	Block entropy	[BDS19]
	Entropy rate	[BDS19]
	Relation entropy	[BvdAŽP11]
Disparity	Average Levenshtein distance	[Pen03]
	Similarity of direct-follows relations	[Gün09]
Hybrid	Distance-based entropy	[BDS19]

Table 3.1: Overview of measures associated with process flexibility.

et al. [BvdAŽP11] and Pentland et al. [PLKH20]. Table 3.1 provides an overview of all identified measures. An overview of the specific equations defined for each measure is provided in the appendix C.1.

Flexibility measures according to Pentland

Some initial quantitative measures of process flexibility, which also consider structural process properties, are proposed by Pentland [Pen03]. These measures involve three different measurement approaches based

on: *average Levenshtein distance*, *Lempel-Ziv complexity*, and *deviation from random*.

The ***average Levenshtein distance*** [Pen03] is calculated as the mean of the Levenshtein distances (cf. Sect. 2.1) between all traces in an event log. It therefore considers the disparity (cf. Sect. 2.3) between the traces based on the minimum number of insertions, deletions, or substitutions of activities required for the pairwise adaptations of the traces.

The variety measure based on ***Lempel-Ziv complexity*** [Pen03] is calculated by applying the Lempel-Ziv algorithm [LZ76] to identify repetitive subsequences within the traces. The algorithm iteratively scans through a given sequence of activities and determines at each step whether the scanned subsequence can be generated by (a) either copying an already identified subsequence (stored in a dictionary) or (b) by inserting a new subsequence. Each time a new subsequence is discovered, it gets added to a dictionary, and the measure increases by one. Thus, the measure can be considered as a counting measure (cf. Sect. 2.3) reflecting the number of identified subsequences, without considering their disparity. As pointed out by Pentland [Pen03], when applying this measure to an event log consisting of multiple traces, these traces need to be concatenated to a single sequence. Pentland further provides an example, showing that the way the traces are concatenated can significantly alter the value of the Lempel-Ziv complexity.

Finally, ***deviation from random*** [Pen03] is calculated by comparing the relative frequencies of the observed directly follows relations between activities in an event log, with a completely random occurrence of these relations, i.e., a random Markov process [GR90]. The more the observed process deviates from a random Markov process, the lower the calculated variety value. The measure does therefore not consider the disparity of the traces in an event log, but the relative frequency in which directly follows relations occur.

Structural flexibility measures according to Günther

Günther [Gün09] proposes several event log based measures to assess the structure of a process in flexible environments, i.e., environments requiring flexible process execution. In contrast to Pentland [Pen03], Günther [Gün09] does not investigate flexibility in order to analyze the performance of a process, but to determine process discovery algorithms and graphical process representations, which are capable of reflecting the flexibility in a comprehensible way. These measures include: *number of events, number of activities, number of traces, activities per trace, completeness of direct-follows relations*, and *similarity of direct-follows relations* [2].

Number of events, ***number of activities***, ***number of traces***, are all additive measures and are simply calculated by respectively counting the number of events, activities, and traces. Günther points out that these measures are especially suited to evaluate the complexity of an event log (cf. the following Sect. 3.1.2) and that they should be considered as complementary measures. However, based on these measures, it is still not possible to assess how events and activities are distributed among the traces and how similar the traces are to each other.

Hence, Günther [Gün09] proposes three more elaborate measures, which also partially consider the structural similarities between traces. ***Activities per trace*** is calculated as the mean number of distinct activities per trace, ***completeness of direct-follows relations*** is calculated as the inverse relation of observed directly follows relations and the number of maximum possible directly follows relations (the square of the number of activities), and ***similarity of direct-follows relations*** is

[2] Günther [Gün09] further proposes a measure considering the time difference between consecutive events and a measure considering the relative importance of activities, which requires a qualitative evaluation from domain experts. These measures are not considered, since they do not reflect the process structure as defined in Sect. 2.1.

calculated as the mean overlap of directly follows relations between all traces in an event log. The first two measures can be considered as counting measures, since they neither consider relative frequencies nor the disparity between the traces. Nevertheless, disparity is considered by the third measure, which includes a pairwise comparison of the traces based on Jaccard coefficient [HKP12].

Entropy-based flexibility measures according to Back et al.

Back et al. [BDS19] propose a set of frequency-related process variety measures based on entropy. Similarly to Günther [Gün09], Back et al. [BDS19] aim to determine suitable process discovery algorithms and graphical process representations, according to the observed flexibility in an event log. The proposed measures can be categorized as: *trace entropy*, *prefix entropy*, *block entropy*, *nearest neighbor entropy*, and *entropy rate*.

Trace entropy [BDS19] simply calculates the entropy (cf. Sect. 2.3.2) over all traces within an event log, where p_i reflects the proportion of trace variants within the set of traces. While this measure reflects the relative frequencies of trace variants, Back et al. [BDS19] point out that it does not reflect the structural disparity between them. For this reason, the authors propose to calculate the entropy over subsequences of the traces, which can be achieved based on prefix or block entropy.

Prefix entropy [BDS19] is calculated as the entropy over all prefixes of the traces within an event log, where the prefixes of a process trace refer to all possible subsequences of a trace, beginning with the start activity. For example, the trace $\sigma_1 = \langle a, b, c \rangle$ consists of the three prefixes $\{\langle a \rangle, \langle a, b \rangle, \langle a, b, c \rangle\}$ [3]. One main shortcoming of this approach is that the measure does not consider structural similarities between

[3] Here the empty set $\langle \varnothing \rangle$ is not considered as a prefix since it is not relevant for the comparison of the traces.

traces that might appear after a distinct prefix. For example, the measure would detect no structural similarity between $\sigma_1 = \langle a, b, c \rangle$ and $\sigma_2 = \langle x, a, b, c \rangle$, since σ_2 contains a different starting activity.

For this reason, Back et al. [BDS19] propose ***Block entropy***, which is calculated as the entropy over the n-grams (cf. Sect. 2.1) of the traces within an event log. The entropy can be either calculated according to the n-grams of a specific length n (***n-block entropy***) or over all n-grams from length one up to the length of the longest trace in the event log (***global block entropy***). Back et al. [BDS19] point out that especially the latter approach is computationally expensive for event logs containing a large number of long traces. Furthermore, block entropy assumes that the analyzed processes are *stationary*, i.e., it does not matter at which position within a trace the identified subsequences occur [BDS19]. However, this assumption might lead to misleading outcomes in cases where the position of the subsequences in a process actually matters. For example, depending on the software development method, it can make a difference whether the verification and validation of a software is conducted at the beginning or at the end of the development process [Som18, AKS+21].

Nearest neighbor entropy [BDS19] allows to overcome the issue of stationarity by additionally considering the edit distance (cf. Sect. 2.1) between traces. Hence, the measure is calculated as entropy over trace clusters [GGPS06, SGvdA09, JCBvdA09c]. The clusters are identified according to the calculated distances between the traces in an event log, and a predefined size k, which defines the number of traces contained in a cluster. To calculate such a measure Back et al. propose to either apply Kozachenko-Leonenko entropy [DF17], which only considers nearest neighbors ($k = 2$), or k^{th} nearest neighbor entropy [SMH+03], which considers clusters of size $k \geq 2$. The measures yield high entropy values if the distances among the traces are high and if they are

evenly distributed among the clusters. Accordingly, nearest neighbor entropy can be considered as hybrid (cf. Sect. 2.3), since it considers the relative frequency as well as the disparity of traces.

Finally, Back et al. [BDS19] propose ***entropy rate***, which is calculated as entropy at the event level, rather than the trace level. Hence, each event is considered as a random variable as part of a stochastic process. To calculate the entropy rate, Back et al. propose three estimators: (a) ratio-based k-block entropy, which is calculated as the ratio of the k-block entropy to the length k, (b) difference-based k-block entropy, which is calculated as difference between the $k+1$-block entropy and k-block entropy, and (c) Lempel-Ziv entropy, which is calculated according to the identified subsequences by the Lempel-Ziv algorithm (similar to *Lempel-Ziv complexity* proposed by Pentland [Pen03]) and the overall number of traces in an event log.

Evolutionary flexibility measures according to Bose et al. and Pentland et al.

To measure the flexibility of a process due to evolution (the fourth flexibility category described by Reichert and Weber [RW12]) requires additional considerations such as the timely occurrence of the change and its effect on the process's performance. These aspects are commonly considered as *concept drift analysis* [BvdAŽP11, vdA16, SDFBS21], whereby concept drift refers to the evolutionary changes of a process, which can be observed in an event log. The timely occurrence of a concept drift can be classified according to four categories [BvdAŽP11, vdA16, SDFBS21]: *sudden drift*, *recurring drift*, *gradual drift*, and *incremental drift*. The different categories are schematically depicted in Fig. 3.1.

However, in each case, concept drift analysis in process mining involves the mapping of the traces in an event log onto a time series, according to their timely occurrence. This can be achieved through

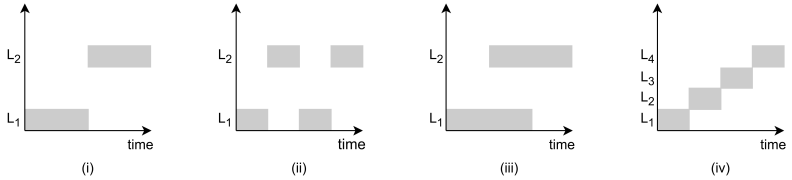


Figure 3.1: Four types of concept drift: (i) *sudden drift*, (ii) *recurring drift*, (ii) *gradual drift*, and (iv) *incremental drift*. L_{1-4} indicating subsets of an event log L. The figure is adapted from Bose et al. [BvdAŽP11].

the application of some *local* or *global* measure, considering respectively process features at a trace or at an event log level [BvdAŽP11]. At a global level, it is thus also possible to consider the evolution of structural process variety, based on the comparison of traces that occur within a predefined time frame. In this regard, Bose et al. [BvdAŽP11] propose the application of ***relation entropy***, which is calculated as the entropy over the *relation type count* for each activity a_i. The relation type count is defined as a triple of numbers representing the activities, which follow after a_i in all traces (always), in a subset of the traces (sometimes), and in none of the traces (never).

Additionally, Pentland et al. [PLKH20] propose the ***number of acyclic traces*** as a measure for concept drift. Similarly to *trace count* [Gün09], the measure counts the number of acyclic execution paths (traces), which can be identified in a process model according to McCabe's cyclomatic complexity [McC76].

3.1.2 Process Complexity

In BPM research, the concept of process complexity is often not clearly distinct from the concept of process flexibility [Men08, Gün09, BWBBG14,

BDS19, PLKH20, AMVW22] [4]. Nevertheless, in contrast to flexibility, complexity analysis in BPM has been commonly linked to the comprehensibility of process models [GL06, MRC07, Men08, SAAW24]. It is thereby generally assumed that the higher the number of elements and constructs in a model and the more interconnected these elements are, the more difficult it is to comprehend. Complexity has therefore often been considered as an additive measure, related to complexity measures proposed in software engineering [GL06, MRC07, Men08, PLKH20] (cf. Sect. 2.3.2).

While there exists a substantial amount of work on the measurement of process model complexity ([GL06, MRC07, Men08]) only two studies conducted by Benner-Wickner et al. [BWBBG14] and Augusto et al. [AMVW22] could be identified, which consider the measurement of event log complexity. Table 3.2 provides an overview of all identified measures. An overview of the specific equations defined for each measure is provided in the appendix C.2.

Complexity measures according to Benner-Wickner et al.

Benner-Wickner et al. [BWBBG14] propose several complexity measures to evaluate the suitability of process discovery. They argue that too high complexity might imply that process discovery is not suitable, since it would result in incomprehensible process models, and there-

[4] Similarly, there appears to be no clear distinction between the concepts of complexity and diversity [BWBBG14, AMVW22]. Benner-Wickner et al. [BWBBG14], for example, use the term *diversity* to refer to complexity. Similarly, Augusto et al. [AMVW22] consider different complexity dimensions, which resemble the three diversity dimensions number, frequency, and disparity. However, a possible distinction is provided by Page [Pag10], who points out that in general the complexity of a system depends on the interdependencies between its entities, e.g., a landfill might not be considered as complex, since the waste products are not inter-dependent. Similarly, the complexity of a software depends on the coupling of its components [BEM96]. This also implies that complexity does not necessarily require diversity, e.g., a computer program relies on a (non-diverse) binary system, enabling the design of highly complex systems [Pag10].

Complexity		
Type	**Name**	**Reference**
Number	Events per trace	[BWBBG14]
	Activities per trace	[BWBBG14]
	Activity event ratio	[BWBBG14]
	Activities per trace ratio	[BWBBG14]
Frequency	Variant entropy	[AMVW22]
	Normalized variant entropy	[AMVW22]
	Sequence entropy	[AMVW22]
	Normalized sequence entropy	[AMVW22]
Disparity	Modified average Levenshtein distance	[BWBBG14]

Table 3.2: Overview of measures associated with process complexity.

fore alternative process analysis techniques should be applied. The following measures are proposed: *events per trace, activities per trace, activity event ratio, activity probability*, and *modified average Levenshtein distance*.

Events per trace and ***activities per trace*** are both calculated by respectively calculating the mean number of events and activities per trace. While *activities per trace* has already been considered by Günther [Gün09], Benner-Wickner et al. [BWBBG14] additionally propose to consider *events per trace*, in order to account for repetitive activities. This is also reflected by the ***activity event ratio***, i.e., the number of activities divided by the number of events. Furthermore, Benner-Wickner et al. [BWBBG14] propose to measure the ***activity per trace ratio*** based on the ratio between *activities per trace* and the overall number of distinct activities in an event log. To also account for the disparity between traces, Benner-Wickner et al. [BWBBG14] define a ***modified average Levenshtein distance***, which is calculated based on the *average*

Levenshtein distance proposed by Pentland [Pen03], and additionally reflects *events per trace* to account for the varying distances between the traces.

Entropy-based complexity measures according to Augusto et al.

Similarly to Günther [Gün09] and Back et al. [BDS19], Augusto et al. [AMVW22] focus on the evaluation of event logs for the identification of a suitable process discovery algorithm. Augusto et al. [AMVW22] consider different complexity perspectives, which relate to the three diversity perspectives: number, frequency, and disparity. Even though these dimensions are recognized, they do not consider a distance measure for the evaluation of complexity. Instead, they propose four entropy-based measures: *variant entropy*, *sequence entropy*, and their corresponding normalized versions. To apply these measures requires at first, a transformation of an event log into a prefix automaton, as proposed by Munoz-Gama and Carmona [MGC10]. The prefix automaton is a directed graph $G(N, E)$ depicting all trace prefixes observed in an event log, where the nodes represent states, and the edges between the nodes represent activities that lead to the respective states. Augusto et al. [AMVW22] further extend the notion of the prefix automaton by associating each node (state) with the events of the traces that lead to the respective state. Subsequently, a partitioning is applied according to the identified prefixes (branches) in the graph. Fig. 3.2 provides an example for an extended prefix automaton and its partitions.

The formulas for the proposed measures by Augusto et al. [AMVW22] are based on the graph entropy measures proposed by Dehner et al. [DM11], which are defined based on the partitioning of a graph. After an extended prefix automaton is derived from an event log, ***variant entropy*** is calculated as entropy over the identified partitions and the

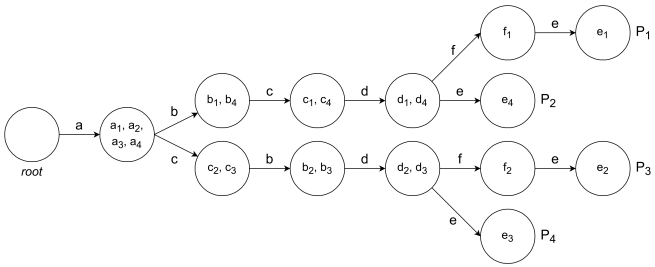


Figure 3.2: Example of an extended prefix automaton for the event log $L = \{\langle a, b, c, d, f, e\rangle, \langle a, c, b, d, f, e\rangle, \langle a, c, b, d, e\rangle, \langle a, b, c, d, e\rangle\}$. The edges of the graph indicate activities, the nodes indicate states. A root node is added to mark the initial state. P_{1-4} refer to partitions of the automaton derived from the prefixes of the traces. The figure is adapted from Augusto et al. [AMVW22].

number of states that belong to the partition. Accordingly, the ***normalized variant entropy*** is calculated based on the overall number of states within the extended prefix automaton. *Sequence entropy* is calculated similarly to ***variant entropy***, but additionally accounts for the number of events associated with each state and the respective partition. Accordingly, the ***normalized sequence entropy*** is calculated based on the overall number of events within the extended prefix automaton.

As pointed out by Back et al. [BDS19] measures based on prefixes fail to detect similarities between traces that occur after common prefixes (cf. Sect. 3.1.1). The measures proposed by Back et al. [BDS19] further fail to consider the disparity between traces, and need to be considered as frequency-based measures reflecting the relative frequencies of prefixes and the associated events.

3.1.3 Exploratory Processes

Exploratory processes can be categorized according to two main objectives: (a) the exploration of information [Mar06, KP17, ZSW21, SAAW23,

SAAW24] and (b) the exploration of a predefined solution space, in order to identify an optimal solution for a defined problem [Urs02, dMWvdA07, Bui14a, MRD+15]. For (a) information exploration, diversity can be considered as a goal in itself, e.g., to provide users with an overview of available options, as in the case of recommender systems [KP17], or available information, as in the case of exploratory search [Mar06, ZSW21, SAAW23, SAAW24]. Especially, for recommender systems there exist a number of diversity measures, proposed in the literature, to guide the selection of recommendations (an overview of generic diversity measures is provided by Kunaver and Požrl [KP17]).

For (b) problems of optimization, diversity can be considered as a guiding principle for the trade-off between exploration and exploitation [Urs02, KP17, Bui14a, MRD+15]. A high diversity of investigated solutions can be associated with exploration, whereas low diversity can be associated with exploitation. The search for an optimal solution is thereby restricted to a predefined solution space, e.g., the discovery of a process model representing a given set of traces [dMWvdA07, Bui14a, MRD+15]. A common approach for solving such an optimization problem is the application of evolutionary algorithms [Urs02, dMWvdA07, MRD+15]. Different diversity measures have been proposed in the literature to guide the search for an optimal solution by evolutionary algorithms [Urs02, LS11, MRD+15].

Table 3.3 provides an overview of all identified diversity measures for (a) information exploration and (b) problems of optimization. An overview of the specific equations defined for each measure is provided in the appendix C.3.

Diversity Measures for Recommender Systems

Diversity measures have been commonly applied to improve the outcome of recommender systems by providing the user with a diverse

Exploration

Type	Name	Reference
Frequency	Gini coefficient	[FH07]
Disparity	Average distance	[CBB13, LCB14]
	Distance to average variant	[Urs02]
Hybrid	Average dissimilarity	[MRD+15]

Table 3.3: Overview of measures associated with exploratory processes.

selection of relevant information or choices (for an overview see Kunaver et al. [KP17]). Even though recommender systems have been extensively investigated in the context of process modeling [KHO11, LCX+13] and process mining [YER22, HJPvdA24], these studies have not considered diversity measures to derive recommendations. One reason for this could be that the existing studies are mainly interested in deriving a single optimal recommendation for the user without considering alternative recommendations, such as the second or third best recommendation. Nevertheless, this approach could also be applied in the context of process recommender systems based on the identified diversity measures by Kunaver and Požrl [KP17].

While several identified measures consider diversity only as an additional aspect of a recommended set of items, next to ambiguity, redundancy, and novelty of the items [KP17], there are two measures specifically focusing on diversity. These are *average distance* [LCB14, CBB13], and *Gini coefficient* [FH07].

Average distance [LCB14, CBB13] resembles the average Levenshtein distance proposed by Pentland [Pen03] and Benner-Wickner et al. [BWBBG14], except that the distance measure is not necessarily specified. The ***Gini coefficient*** [FH07] is a measure for distributional inequality, where zero indicates perfect equality and one indicates max-

imum inequality between variants (see also Han et al. [HKP12] for a more detailed description of its calculation). Fleder and Hosanagar [FH07] propose to use this measure to analyze the effect of product-related recommender systems on the diversity of sold products. Similarly, this measure could, for example, be used to evaluate the contribution of recommended process model elements to the overall process model [KHO11, LCX+13].

Diversity Measures for Evolutionary Algorithms

Similarly, diversity has been recognized as a guiding principle for process discovery based on evolutionary algorithms [Bui14a, MRD+15]. Evolutionary process discovery seeks to identify an optimal process model representation for an event log, according to some predefined quality measures. The evolutionary process of this process discovery approach is depicted in Fig. 3.3. The general idea is to generate a population of distinct process models, which are iteratively modified (*mutated*) and combined (based on *crossover*) to derive new populations of better quality (according to some *fitness* value). Diversity measures can thereby help to control the exploration of the solution space to avoid premature convergence to suboptimal solutions [Urs02].

Several general heuristics have been proposed in the literature to ensure the diversity of a population without relying on a diversity measure (for an overview see Gupta and Ghafir [GG12], and Cully and Demiris [CD17]), such as the introduction of randomization in the generation of populations. However, there also exist heuristics relying on diversity measures [Urs02, LS11, MRD+15]. These measures all relate to the *average distance* between variants and include: *distance to average point* [Urs02], *average distance to k-nearest neighbors* [LS11], and *sum of distance to average point* [MRD+15].

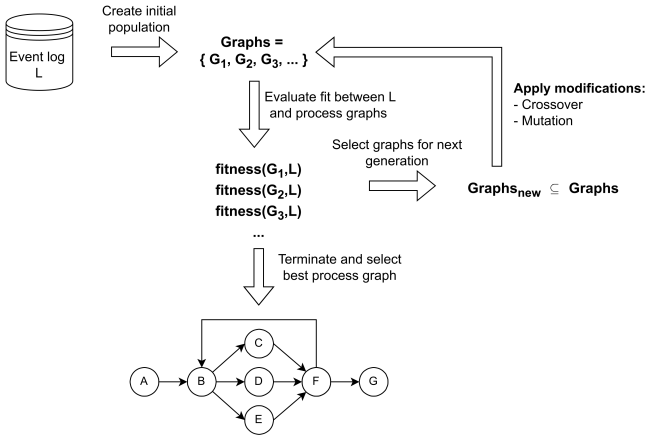


Figure 3.3: Schematic depiction of evolutionary process mining based on [vdA16].

Distance to average point [Urs02] is calculated as the average of the distance between a variant and the average over all variants, across all dimensions of a search space, e.g., also considering process perspectives beyond the structure, such as involved resources or durations. Similarly to the measures proposed by Kozachenko and Leonenko et al. [DF17] (cf. Sect. 3.1.1), the ***average distance to k-nearest neighbors*** considers the distances between a variant and its k-nearest neighbors, where k is a predefined fixed value. The ***sum of distance to average point*** [MRD+15] is the only measure specifically proposed in the context of evolutionary process discovery. For this measure Molka et al. [MRD+15] propose to use the Manhattan distance (a description for the calculation of this distance can be found in Han et al. [HKP12]) in order to calculate the sum of distances over all variants and a mean variant. The mean variant is calculated based on the relative frequencies of the directly follows relations, observed over all traces. Hence, the measure can be considered as a hybrid measure considering the

disparity between the traces (based on Manhattan distance) as well as the average frequencies of directly follows relations.

3.2 Requirements for Measuring Structural Process Variety

The study of the related work reveals that structural process variety has been considered for the analysis of *process flexibility*, *process complexity*, and *exploratory processes* (cf. Sect. 3.1). Based on the identified measures, one can derive the following conclusions, which reveal four general requirements for the measurement of structural process variety.

(1) Relevance of the Diversity Dimensions

In all three types of analysis (flexibility, complexity and exploration), different dimensions of diversity have been considered. A majority of the measures can be categorized as counting measures (11) and frequency-related measures (11). Fewer measures can be categorized as disparity measures (6) and hybrid measures (2). The need to consider the different dimensions of diversity for the analysis of structural process variety has been recognized by Back et al. [BDS19] and Augusto et al. [AMVW22]. Yet, both studies only propose entropy measures, which do not allow for a separation or weighting of the distinct dimensions. This increases the complexity of the measures and makes them difficult to interpret.

An example for this is the *distance-based entropy* measure proposed by Back et al. [BDS19]. The measure combines the relative frequency of traces and the distance between them. However, it also depends on a parameter describing the dimension of the metric space ($d = 1$) and two constants (π and Euler's constant) for which Back et al. do not provide an interpretation in the context of the variety evaluation (the

complete formula is described in the appendix C.1). Thus, it is difficult to assess why a particular event log has a high or low variety value according to the proposed measure.

Furthermore, most studies consider different diversity dimensions to measure the same phenomena, i.e., flexibility [Pen03, Gün09, BDS19], or complexity [BWBBG14, AMVW22]. As shown in Sect. 2.3.2, this can lead to contradictory outcomes. A clear distinction of the dimensions can help to avoid such contradictions and to gain better insights into the analyzed phenomena [Sti07, DBDB18].

As a conclusion, it can be assumed that the concept of diversity, relying on the three different dimensions number, frequency and disparity, as proposed in the literature [Bau06, Sti07, Gra08, NP09] can be meaningfully applied in the context of process analysis. Accordingly, it can also be assumed that the diversity measures, proposed in the literature (cf. Sect. 2.3), can provide meaningful insights for the analysis of structural process variety.

(2) Structural Trace Properties

The need to reflect structural properties of the traces for the analysis of process variety has been acknowledged across different process analysis techniques [Pen03, Gün09, BWBBG14, MRD+15, BDS19, AMVW22]. Different approaches for the structural comparison of traces have been proposed in the literature. Yet, only three different distance measures have been considered to account for the disparity between traces, i.e., Levenshtein distance [Pen03, BWBBG14, BDS19][5], Jaccard coefficient [Gün09], and Manhattan distance [MRD+15].

However, none of the identified studies considers multiple different distance measures to analyze their impact on the variety evaluation.

[5] Both Benner-Wickner et al. [BWBBG14] and Back et al. [BDS19] consider a normalized version of the Levenshtein distance to account for repetitive behavior (loops) in the traces.

Since different distance measures address different structural properties of the traces [BS23], one would expect that they would also have a significant impact on the variety evaluation. In a comprehensive study, Back and Simonsen [BS23] provide a comparison of 18 different distance measures for the comparison of trace similarity. The study shows that the distance measures yield significantly different outcomes when applied to the same set of traces.

To further elaborate on the role of the distance measure for the evaluation of structural process variety, the following Chapter 4 provides a comparison of different distance measures, which investigates their ability to reflect different structural trace properties. Based on the comparison, an alternative measurement approach is introduced, which allows to also consider higher-level structural trace properties.

(3) Mathematical Properties

Regarding mathematical properties of structural process variety measures, Back et al. [BDS19] discuss general mathematical properties of Shannon-Wiener entropy, i.e., continuity, monotonicity, and decomposability. However, they do not discuss their relevance for the evaluation of structural process variety. Similarly, Augusto et al. [AMVW22] consider the relevance of monotonicity with respect to the increase in the number of distinct variants for the evaluation of complexity. Other mathematical properties of Shannon-Wiener entropy, i.e., symmetry, maximum of equality, and scale invariance (which have been discussed in the literature on diversity measures, cf. Sect. 2.3.2) are not considered.

For the identified disparity-based measures, none of the respective studies provide any considerations of mathematical properties. This is especially critical, since all of the disparity measures, i.e., *average Levenshtein distance* [Pen03], *similarity of directly follows relations* [Gün09], *mod-*

ified average Levenshtein distance [BWBBG14], *average distance* [LCB14, CBB13], *distance to average variant* [Urs02], and *average distance to k-nearest neighbors* [LS11] rely on the calculation of the mean distance between traces. This could be considered critical, since the arithmetic mean violates monotonicity with respect to an increase in the number of trace variants. Additionally, the arithmetic mean of the trace distances is discontinuous when the distances decrease and merge in the limits [NP09]. However, both monotonicity and continuity have been considered relevant mathematical properties of a measure when it comes to the evaluation of diversity across different domains [Sti07, Gra08, NP09, DBDB18] and also for the evaluation of process variety [BDS19, AMVW22].

An additional critical point of the identified structural variety measures is that they do not allow for a ranking of the traces according to their marginal contribution to the overall variety of the set [Wei92, NP02, Gra08]. However, this can be particularly relevant in practice, e.g., if one seeks to evaluate the impact of a single trace variant on the overall complexity of a process [GL06, Men08], or in the context of business process standardization [SAA24].

The following Sect. 3.3 provides a framework, which allows measuring structural process variety of an event log by considering (only) the disparity between its traces. The framework follows from the distance-based measurement approach proposed by Weitzman [Wei92] and guarantees monotonicity. It further provides a ranking of the traces according to their contribution to the overall variety of the event log.

(4) Computational Complexity

Finally, several of the identified studies [Pen03, BWBBG14, BDS19, AMVW22] consider computational complexity as a critical aspect for the development of a structural process variety measure. This accounts

especially for measures, which involve a pairwise comparison of traces [Pen03, BDS19]. High computational complexity can limit the practical use of the measure in real-world scenarios involving large event logs, such as in business process analysis [BDS19, BS23]. Hence, balancing computational complexity with the expressiveness (involving relevant structural properties) and the accuracy (involving relevant mathematical properties) of the structural process variety measure is essential to ensure scalability without compromising the quality of the variety assessment.

Accordingly, the computational complexity will also be considered as an evaluation criteria for the measurement framework introduced in the following Sect. 3.3 and the trace similarity measures introduced in Sect. 4.3.

3.3 Measuring Structural Process Variety

Based on the identified research gaps and requirements from the analysis of the literature in the previous Sect. 3.2, this section considers the application of the measurement approach proposed by Weitzman [Wei92] for the evaluation of structural process variety. The measurement approach allows focusing on the disparity between trace variants, without considering their number or frequency. Accordingly, the structural process variety is calculated based on a flat event log, i.e., a set of trace variants. First, a distance matrix is calculated for a given flat event log $L' = [\sigma_1, ...\sigma_m]$, which defines a pairwise distance $d(\sigma_i, \sigma_j)$ for all traces variants $\sigma_i, \sigma_j \in L'$. Subsequently, the distance between a trace variant $\sigma_i \in L'$ and a subset of trace variants $Q \subset L'$ can be calculated as follows:

$$\delta(\sigma_i, Q) = \min_{\sigma_j \in Q} d(\sigma_i, \sigma_j) \text{ for all } \sigma_i \in L' \setminus Q. \tag{3.1}$$

To derive the overall structural variety value $V_{\text{Weitzman}}(L')$ one needs to iteratively add up all distances $\delta(\sigma_i, Q)$, starting with some arbitrary subset $Q = \{\sigma_{j\neq i}\} \in L'$ and starting value $V_{\text{Weitzman}}(\{\sigma_{j\neq i}\})$:

$$V_{\text{Weitzman}}(Q \cup \{\sigma_i\}) = V_{\text{Weitzman}}(Q) + \delta(\sigma_i, Q) \quad \text{for all } \sigma_i \in L' \setminus Q \text{ and } \varnothing \subset Q \subset L'. \tag{3.2}$$

With respect to structural process variety, it is reasonable to assume that the initial value for $V_{\text{Weitzman}}(\{\sigma_{j\neq i}\})$ is zero, as there is no variety when L' contains only a single trace variant. According to Equation (3.2), the variety value will then incrementally increase as the remaining trace variants $\sigma_i \in L' \setminus Q$ are added to Q. The variety increases each time based on the distance $\delta(\sigma_i, Q)$, as specified in Equation (3.1).

As discussed in Sect. 2.3.2, ensuring the uniqueness of V_{Weitzman} necessitates $d(\sigma_i, \sigma_j)$ to be an ultrametric [Wei92, NP02]. An ultrametric in this case is a metric with the additional property that the two largest distances among any three trace variants $\sigma_i, \sigma_j, \sigma_k \in L'$ are always equal. For example, if $d(\sigma_1, \sigma_2) = 1$ and $d(\sigma_2, \sigma_3) = 2$, then it follows that $d(\sigma_1, \sigma_3) = 2$. This condition can be satisfied through the application of *hierarchical trace clustering* [GGP05, FZMF07, SGvdA09, JCBvdA10]. Hierarchical clustering ensures that the distances between the resulting clusters are ultrametric, as proven by Johnson [Joh67]. Therefore, to derive a unique structural variety value $V_{\text{Weitzman}}(L')$, one must first group the trace variants using hierarchical clustering and then calculate the variety based on the ultrametric distances between the clusters.

Hierarchical clustering has been initially introduced in process mining, since the hierarchical structure of the clusters can help to analyze different abstraction levels of a process [GGP05]. The different structural properties of a process at the different levels can accordingly be de-

scribed by a process model taxonomy. Fig. 3.4 provides an example of such a taxonomy, which is depicted as a tree consisting of multiple subprocess. The root of the tree encodes the most abstract view of the process, and each following node represents a refinement of the process.

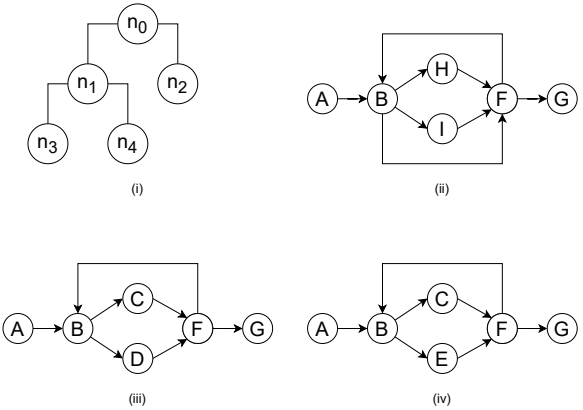


Figure 3.4: (i) represents a (taxonomic) tree indicating the hierarchical relationships of the subgraphs n_{0-4}, (ii) represents the subgraph n_2, (iii) represents the subgraph n_3, and (iv) represents the subgraph n_4. The example is based on [GGP05].

Additionally, the mathematical properties of the distance-based measure (cf. Sect. 2.3.2) ensure a meaningful evaluation of structural process variety and are in accordance with the identified requirements in Sect. 3.2, i.e.:

1. $V_{\text{Weitzman}}(L')$ *monotonically increases* with respect to the number of trace variants
2. $V_{\text{Weitzman}}(L')$ *monotonically increases* with respect to the pairwise distances between the trace variants
3. $V_{\text{Weitzman}}(L')$ is *continuous* with respect to the pairwise distances between the trace variants in L'

4. the extraction of some trace variant σ_i from L' will reduce the diversity of L' by exactly the distance of σ_i and another remaining trace variant in $\sigma_j \in L'$ (*linking property*), thereby allowing to derive a *ranking* of the traces
5. the diversity does not increase when an already existing trace variant is added to L' (*twin property*), i.e., the measure only considers the disparity between trace variants without their relative frequencies
6. the upper bound for $V_{\text{Weitzman}}(L' \cup \sigma_i)$ with $\sigma_i \notin S$ is the maximal distance $d(\sigma_i, \sigma_j)$ for some $\sigma_j \in L'$.

The application of the iterative measurement approach proposed by Weitzman [Wei92] to evaluate structural process variety requires several steps. The following section introduces a framework, which allows to systematically derive $V_{\text{Weitzman}}(L')$ and to visualize the respective ranking of the traces.

3.3.1 Structural Variety Measurement Framework

The framework to derive $V_{\text{Weitzman}}(L')$ according to Equation 3.2 is depicted in Fig. 3.5. The framework consists of five consecutive steps: (1) *preprocessing*, (2) *calculating the distance matrix*, (3) *applying hierarchical clustering*, (4) *calculating variety*, and (5) *visualization*. Steps (2)-(4) define the core of the framework that is necessary to actually derive $V_{\text{Weitzman}}(L')$. (1) *Preprocessing* and (5) *visualization* can be considered optional, depending on the available data and the goal of the analysis. In the following, each step is described in detail.

Preprocessing

Preprocessing can itself encompass multiple steps, such as the finding, merging, and cleaning of event data [vdA12]. As stated in the pro-

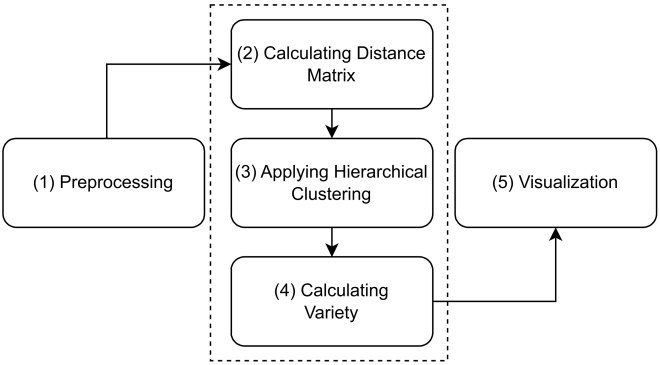


Figure 3.5: Proposed framework for the measurement of structural process variety.

cess mining manifesto [vdA12], process data may be distributed over a variety of sources. Hence, *finding* and *merging* this data to derive a consistent event log can be an essential preprocessing task.

Furthermore, the event log itself might require additional *cleaning*. This might involve the detection of *missing data*, *incorrect data*, *imprecise data*, and *irrelevant data* [BMvdA13]. The identification of these data issues depends, of course, on the context and the goal of the analysis. Here it is particularly important to consider data issues that might impact the structural variety evaluation. Possible questions that can guide the preprocessing in this regard include:

- are all relevant trace variants included in the event log (relates to *missing data*)
- does the data correctly represent the sequence of events or are events eventually dated incorrectly (relates to *incorrect data*)
- does the data represent the right level of granularity, which the structural variety analysis aims at (relates to *imprecise data*)

- are the structural differences between the traces actually representative for the structure of the process, or can they be considered as noise (relates to *irrelevant data*).

To apply the framework to a given event log L, the preprocessing phase also includes the identification of the trace variants, i.e., $f_{flat}(L) = L' = [\sigma_1, ..., \sigma_m]$ (cf. Sect. 2.1).

Calculating Distance Matrix

In this step, a distance function d is selected and applied to determine the pairwise distances for every $\sigma_i, \sigma_j \in L'$. The distance measure thereby seeks to identify the structural similarities and differences between the traces. Several such distance measures have been proposed and analyzed in the literature (for an overview see Back and Simonsen [BS23]). Depending on the structural properties that are considered for the comparison, the existing measures can differ significantly in their evaluation of the distances between traces. For this reason, Back and Simonsen [BS23] propose a set of evaluation measures, which can help to select a fitting distance function for a specific log. An overview of the considered evaluation measures in this thesis can be found in the appendix B.

One shortcoming of the evaluation measures is that they require that the trace variants in L' can be clearly distinguished from one another based on some defined class labels. Furthermore, they assume that the distances among traces within the same classes are lower than the distances among traces belonging to different classes. Depending on the context, these assumptions can be questionable, since it might not always be possible to provide such a classification of traces. To overcome this issue, it is also possible to consider evaluation measures which do not rely on labeled data, i.e., a predefined classification of the trace variants. This is, for example, the case for the *silhouette* measure [BS23].

L'
$\sigma_1 = \langle A, B, C, D, E, F, G, H \rangle$
$\sigma_2 = \langle A, B, C, E, F, G, H, D, I \rangle$
$\sigma_3 = \langle A, B, C, E, F, G, H, D \rangle$
$\sigma_4 = \langle A, B, C, E, D, F, G, H \rangle$
$\sigma_5 = \langle A, B, C, E, F, D, G, H, I \rangle$

$$\mathbf{D_{L'}} = \begin{bmatrix} 0 & 3 & 2 & 2 & 3 \\ 3 & 0 & 1 & 3 & 2 \\ 2 & 1 & 0 & 2 & 2 \\ 2 & 3 & 2 & 0 & 3 \\ 3 & 2 & 2 & 3 & 0 \end{bmatrix}$$

Figure 3.6: An illustrative flat event log L' and the calculated distance matrix $D_{L'}$ based on Levenshtein distance. The example is taken from Bose and van der Aalst [JCBvdA10].

However, in this thesis only event logs with classified traces are considered for the evaluation.

Furthermore, it is possible to compare the distance measures based on some benchmark data set, where the structural differences between the traces are well determined. Such benchmark data set is provided in the following Ch. 4, which also provides a more elaborate discussion on the selection and evaluation of suitable similarity measures for the analysis of structural similarities and differences between traces.

As discussed in Sect. 2.1 a common distance measure applied in process mining is the Levenshtein distance [Pen03, BvdAŽP11, BDS19, BS23]. The Levenshtein distance [Lev66] between two traces is determined by counting the number of insertions, deletions, and substitutions of activities that are necessary to change one trace into the other. Fig. 3.6 provides an example for the calculation of a distance matrix D for a log L_0 based on Levenshtein distance.

Applying Hierarchical Clustering

Hierarchical clustering has been extensively explored in the context of process mining [GGP05, FZMF07, SGvdA09, JCBvdA10]. The hierarchical clustering approach appears particularly suited for clustering traces due to its ability to capture the inherent hierarchical structure of a process [GGP05, FZMF07] (cf. Sect. 3.3). The structural properties

can thereby be categorized as *global* and *local* [SGvdA09, JCBvdA10]. *Global* structural properties can be identified in a broad subset of traces, such as initial and terminal sequences. Whereas *local* structural properties pertain to smaller subsets of traces, like the recurrence of specific sequences of activities.

Hierarchical clustering algorithms can be classified into two primary methods: *agglomerative* and *divisive clustering* [HKP12]. *Agglomerative clustering* starts with each trace as an individual cluster and iteratively merges the closest pairs of clusters until a single cluster encompassing all traces is formed. Conversely, *divisive clustering* begins with a single cluster encompassing all traces and recursively splits the traces into smaller clusters based on the dissimilarities between traces until each trace forms its own cluster.

The outcome of the agglomerative and the divisive clustering depends in both cases on the defined *linkage criterion*, which determines how the distance between clusters is measured [HKP12]. There exist three linkage criteria, which are commonly applied in hierarchical clustering [FG08, HKP12]: *single linkage*, *complete linkage*, and *average linkage*.

Given a defined distance measure $d(\sigma_i, \sigma_j)$ and two clusters $C_i, C_j \subseteq L'$, such that $\sigma_i \in C_i$ and $\sigma_j \in C_j$, then the linkage criteria can be defined as follows:

Single linkage: calculates the distance between two clusters as the smallest distance between any pair of traces, consisting of one trace from each cluster.

$$d_{min}(C_i, C_j) = \min d(\sigma_i, \sigma_j) \text{ for all } \sigma_i \in C_i \text{ and } \sigma_j \in C_j \tag{3.3}$$

Complete linkage: calculates the distance between two clusters as the largest distance between any pair of traces, consisting of one trace from each cluster.

$$d_{max}(C_i, C_j) = \max d(\sigma_i, \sigma_j) \text{ for all } \sigma_i \in C_i \text{ and } \sigma_j \in C_j \tag{3.4}$$

Average linkage: calculates the distance between two clusters as the average distance between any pair of traces, consisting of one trace from each cluster.

$$d_{avg}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\sigma_i \in C_i} \sum_{\sigma_j \in C_j} d(\sigma_i, \sigma_j) \tag{3.5}$$

There also exists a variety of alternative linkage criteria. However, these criteria might require a transformation of the traces into a feature space, as is the case for *centroid linkage*, which relies on Euclidean distance [FG08, HKP12].

An illustrative example of hierarchical clustering based on the average linkage criterion applied to the flat event log L' (from the example in Fig. 3.6) is presented in Fig. 3.7.

Similarly to the selection of a distance measure (cf. Sect. 3.3.1) there exist several evaluation criteria, which can be used to evaluate the suitability (or quality) of a selected clustering algorithm and its parameters for a given event log. These criteria can be classified as: *extrinsic criteria*, which compare the derived clustering against a ground truth, i.e., the actual trace clusters (or assigned class labels) are known, and *intrinsic criteria*, which evaluate the quality of the derived clustering by considering how well the clusters are separated from one another and how compact they are [HKP12]. Han et al. [HKP12] identify four essential *extrinsic criteria*:

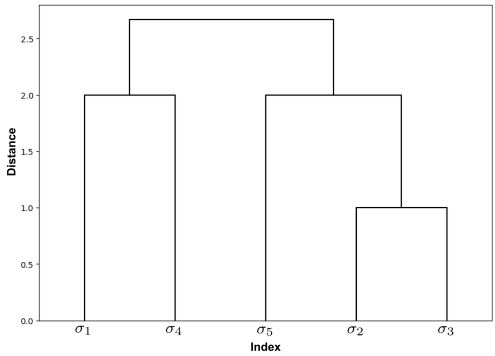


Figure 3.7: The dendrogram derived from the hierarchical clustering of the trace variants in L′ and their pairwise distances $D_{L'}$.

- ***Cluster homogeneity***: considers how homogeneous the traces are within a cluster according to their class labels. The more traces are from the same class, the higher the homogeneity of this cluster.
- ***Cluster completeness***: considers whether traces with an identical class label are assigned to the same cluster. The more traces of the same class are assigned to the same cluster, the higher the completeness of the clusters.
- ***Ragbag***: in some cases, event logs might contain traces that do not fit well with other classes, as they might represent exceptional behavior. A clustering algorithm should detect these traces as ”ragbag" and not place them with some homogeneous cluster. The more traces are placed as ”ragbag" rather than disrupting the homogeneity of another cluster, the higher the ragbag.
- ***Small cluster preservation***: considers the preservation of the integrity of classes consisting of a relative small number of traces in a clustering. This criterion states that splitting a small class is more detrimental than splitting a large class.

Han et al. [HKP12] further identify one essential *intrinsic criteria*:

- ***Silhouette coefficient***: considers both the compactness of a cluster, based on the average distance among the traces within a cluster, and the separation of the clusters, based on the average distance between the traces in a cluster and their distance to the nearest neighboring cluster. A high silhouette coefficient signifies compact and well-separated clusters, while a low coefficient suggests poor clustering where traces may be misplaced.

Moreover, hierarchical clustering algorithms can be visualized using dendrograms, which graphically represent the merging or splitting process and the resulting nested sequence of clusters (as depicted in Fig. 3.7). This visualization provides insights on the hierarchical relationships between traces and the structural properties of the process. The dendrogram can therefore also help to evaluate the quality of the applied clustering algorithm, assuming that a domain expert can validate its structure.

It is important to acknowledge that hierarchical clustering introduces a degree of uncertainty in the measurement of structural process variety based on $V_{\text{Weitzman}}(L')$. This uncertainty arises from the variability inherent in the algorithmic approach and instances where multiple distances between different clusters coincide during the clustering process [FG08]. These issues can be partially mitigated by identifying the optimal hierarchical clustering according to the introduced quality criteria.

The issue of tie-breaking, due to coinciding distances between different clusters during the clustering process, is typically resolved using arbitrary criteria, which may result in non-unique outcomes [FG08]. Fernández and Gómez [FG08] propose an alternative solution that involves grouping more than two clusters simultaneously when ties occur, leading to a multivalued tree with lower and upper bounds at which traces can be merged into a specific cluster. Consequently,

the calculation of $V_{\text{Weitzman}}(L')$ would encompass a lower and upper bound, reflecting the level of uncertainty in the variety evaluation.

Calculating Variety

Given that hierarchical clustering ensures an ultrametric distance between trace variants (cf., Sect. 3.3), a unique structural variety of the log can be computed based on the derived dendrogram. The recursive formula 3.2 is applied using the calculated linkage distances between the traces and their respective clusters. The newly calculated distances based on $D_{L'}$ are illustrated in Fig. 3.8.

$$\mathbf{D'_{L'}} = \begin{bmatrix} 0 & \frac{8}{3} & \frac{8}{3} & 2 & \frac{8}{3} \\ \frac{8}{3} & 0 & 1 & \frac{8}{3} & 2 \\ \frac{8}{3} & 1 & 0 & \frac{8}{3} & 2 \\ 2 & \frac{8}{3} & \frac{8}{3} & 0 & \frac{8}{3} \\ \frac{8}{3} & 2 & 2 & \frac{8}{3} & 0 \end{bmatrix}$$

Figure 3.8: New distance matrix $D'_{L'}$ derived from the outcome of the hierarchical clustering.

Given the dendrogram in Fig. 3.7 and the derived distances between the clusters (depicted in Fig. 3.8) $V_{\text{Weitzman}}(L') = 1 + 2 + \frac{8}{3} + 2 = \frac{23}{3}$. The distances refer to the minimum distances between σ_2 and σ_3, $\{\sigma_2, \sigma_3\}$ and σ_5, $\{\sigma_2, \sigma_3\, \sigma_5\}$ and σ_4, and $\{\sigma_2, \sigma_3\, \sigma_5\, \sigma_4\}$ and σ_1.

Finally, one can visualize the marginal contribution of individual traces to the overall structural variety of the log, as per the recursive formula 3.2.

Visualization

As can be seen by the calculation of $V_{\text{Weitzman}}(L')$, the ranking of the traces is not unique, due to the ultrametric property of the distance measure and the equality of the distances between the traces. Nevertheless, it is possible to derive a structural variety profile, which indicates the distribution of the structural variety among the trace variants within a log.

For practical reasons as will be discussed in Sect. 5 the visualization represents the decrease in structural variety, by iteratively removing a trace variant from the event log. Rather than the increase of variety, by iteratively adding a trace variant to the event log. However, both visualizations would be possible.

If each trace variant contributes equally to the structural variety, the reduction in variety would exhibit a linear pattern. Conversely, if the contributions are distributed unequally, the reduction in variety would display a concave shape. In the example illustrated in Fig. 3.9, all trace variants contribute relatively equally to the log's structural variety, resulting in a variety curve that follows a linear trajectory.

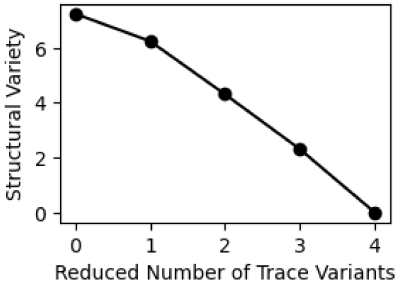


Figure 3.9: A visualization of the decrease in variety according to the incremental reduction of trace variants from L'. The trace variants are removed based on their minimum distances.

3.3.2 Evaluation

To evaluate the proposed measurement framework, seven artificial event logs are considered, which each possess a distinct set of structural properties. Furthermore, the event logs can be categorized according to three different levels of structural variety (low, medium, and high). This allows to compare the framework's outcomes with alternative variety measures proposed in the research literature and determine whether these measures accurately evaluate structural variety across different logs. The logs vary based on the number of trace variants and the type of structural properties, such as ordering, recurrence, and skipping of activities. A similar evaluation approach was employed by Back et al. [BDS19].

Based on the evaluation, we aim to answer the following main question:

- Can the proposed measurement framework consistently evaluate the structural variety across different event logs with distinct structural properties?

The data and implementations for the evaluation are available via the following repository:
https://github.com/promilab/StructuralProVar.git. The implementations are Python-based and utilize the pm4py library [BvZS23].

An additional evaluation of the framework based on real-world business process data follows in Sect. 5. This evaluation considers the applicability of the framework in the context of business process standardization, which is a common BPM practice [MEW10, Tre14, FAM18]. The goal of this evaluation is to show that also the ranking of the traces according to the framework can help to balance process efficiency and flexibility, which is a common trade-off in business process standardization.

Process Data and Implementation

For the evaluation of the consistency of the measure, seven artificial log files are considered. The first three logs are presented in Table 3.4. Each trace variant in these logs comprises five activities, including a fixed start and a fixed end activity. The logs differ based on the following properties: in L_1 the activities (B, C, D) occur in any possible order, in L_2 the activities (B, C, D) occur in a fixed order, but in each trace variant at least one activity has to occur twice, and in L_3 the activities (B, C, D) occur in a fixed order, but in each trace variant at least one activity is skipped.

Each trace variant in these logs consists of five activities, including a fixed start and a fixed end activity. The logs differ based on the following properties: in L_1 the activities (B, C, D) occur in any possible order, in L_2 the activities (B, C, D) occur in a fixed order, but in each trace variant at least one activity has to occur twice, and in L_3 the activities (B, C, D) occur in a fixed order, but in each trace variant at least one activity has to be skipped. These three logs represent different types of structural properties relevant for structural variety evaluation, namely, the ordering, recurrence, and skipping of activities [Pen03, BDS19, TLRDM21]. By incrementally combining the logs as follows:

- $L_4 = L_1 \cup L_2$,
- $L_5 = L_2 \cup L_3$,
- $L_6 = L_3 \cup L_1$,
- $L_7 = L_1 \cup L_2 \cup L_3$,

four additional logs are derived, with an increased number of trace variants and structural properties. The structural differences between the seven logs allow for the evaluation of the consistency of the mea-

sures based on their monotonicity with respect to the increase in structural variety, i.e., $V(L') < V(L' \cup \sigma_i)$ where $\sigma_i \notin L'$.

L_1	L_2	L_3
$\langle A, B, C, D, E \rangle$	$\langle A, B, B, C, D, E \rangle$	$\langle A, C, D, E \rangle$
$\langle A, B, D, C, E \rangle$	$\langle A, B, C, C, D, E \rangle$	$\langle A, B, D, E \rangle$
$\langle A, C, B, D, E \rangle$	$\langle A, B, C, D, D, E \rangle$	$\langle A, B, C, E \rangle$
$\langle A, D, B, C, E \rangle$	$\langle A, B, C, B, C, D, E \rangle$	$\langle A, D, E \rangle$
$\langle A, C, D, B, E \rangle$	$\langle A, B, C, D, C, D, E \rangle$	$\langle A, B, E \rangle$
$\langle A, D, C, B, E \rangle$	$\langle A, B, C, D, B, C, D, E \rangle$	$\langle A, E \rangle$

Table 3.4: Three artificial logs with distinct structural properties.

Structural Variety Comparison

To evaluate how well the measure V_{Weitzman} reflects the overall structural variety of a log, V_{Weitzman} is calculated for the seven artificial logs $L_1 - L_7$ based on the introduced framework. The structural differences of the logs, in terms of number of traces and structural properties, should be reflected by the derived values. Additionally, the results are compared with five other variety measures identified in the literature (cf. Sect. 3.1). These measures are selected according to their ability to consider structural properties. The first four measures proposed by Back et al. [BDS19] are based on entropy and involve different types of trace abstractions, while the fifth measure simply takes the average Levenshtein distance over all traces in a log:

- $V_{\text{N-block}}$ - entropy based on n-blocks (n=2) [BDS19],
- $V_{\text{Lempel-Ziv}}$ - entropy based on Lempel-Ziv complexity [BDS19],
- V_{Prefix} - entropy based on prefixes [BDS19],
- V_{Knn} - entropy based on k Nearest Neighbors and normalized Levenshtein distance (k=1) [BDS19],

- $V_{\text{Levenshtein}}$ - average Levenshtein distance [Pen03].

The results of the calculated variety values are presented in Table 3.5. It is evident that the structural variety measure V_{Weitzman}, based on the proposed framework, is the sole measure that consistently evaluates the variety of the logs in relation to the increase in the number of trace variants and structural properties. Other measures fail to consistently evaluate the variety across the logs due to their sensitivity to the relative frequencies of the structural properties; for instance, certain structural properties, such as the relative ordering of activities in L_2 and L_3, occur more frequently than others. Moreover, the variety values derived from V_{Weitzman} offer greater interpretability compared to other computed measures, enabling a clear differentiation in structural variety among small-sized logs (L_1, L_2, L_3), medium-sized logs (L_4, L_5, L_6), and the largest log (L_7).

M.	L1	L2	L3	L4	L5	L6	L7
$V_{\text{N-block}}$	3,59	2,76	3,13	**3,47**	3,21	3,60	**3,53**
$V_{\text{Lempel-Ziv}}$	2,29	1,86	2,16	**2,01**	**2,07**	**2,15**	**2,09**
V_{Prefix}	4,19	4,14	3,31	4,78	4,38	4,24	**4,78**
V_{Knn}	2,15	1,05	1,72	**2,13**	2,08	2,28	2,50
$V_{\text{Levenshtein}}$	1,67	1,17	1,17	2,00	2,25	1,71	**2,20**
V_{Weitzman}	14,14	12,33	12,33	40,40	44,96	38,86	77,45

Table 3.5: Comparison of different log variety measures (M.) based on seven different logs with low structural variety ($L_1 - L_3$), medium structural variety ($L_4 - L_6$), and high structural variety (L_7). Violations of monotonicity are marked in bold.

3.4 Discussion

While the ultrametric property ensures the uniqueness of a structural process variety measure by enforcing strict hierarchical relationships

among trace variants, it also introduces a significant limitation. Specifically, this property imposes a highly restrictive condition on the structure of the data, which may not be satisfied in many real-world scenarios. In particular, constructing a meaningful hierarchical clustering is not always feasible, especially when the set of trace variants lacks a coherent or interpretable taxonomy of process patterns.

Moreover, the proposed solution based on hierarchical clustering exhibits additional methodological constraints. Although it provides a structured way to analyze variety, its effectiveness is contingent on the quality of the underlying similarity measures and the structural features considered. Without careful refinement, the clustering may fail to reflect meaningful structural distinctions. To address this, the thesis suggests enhancing robustness and validity by incorporating richer structural descriptors and refining similarity measures as described in the following Chapter 4. Furthermore, alternative deterministic clustering techniques, such as those based on deterministic annealing [HB02], are proposed as promising avenues for overcoming the limitations of traditional hierarchical approaches.

A third limitation concerns the non-uniqueness of trace rankings within the proposed framework. Due to the ultrametric constraint, the maximum pairwise distance among any three variants is always equal, which prevents the establishment of a unique global ranking of all traces. Nevertheless, the framework enables the ranking of trace pairs, offering a partial but informative view of the distribution of structural variety. This pairwise ranking, when combined with the proposed visualization techniques, supports the identification of structurally diverse regions within the process landscape and facilitates interpretability.

3.5 Summary

The concept of structural process variety is closely related to several well-established concepts in BPM and process mining, including process flexibility, process complexity, and exploratory processes. While a number of process variety measures have been proposed in the literature, most focus primarily on relative frequency distributions of trace variants or on hybrid approaches that combine frequency with disparity [BDS19, AMVW22]. However, only a limited subset of these measures directly addresses the structural disparity between traces. Moreover, existing variety measures often suffer from mathematical and interpretive limitations. In particular, they typically do not support a meaningful decomposition or ranking of individual trace variants based on their contribution to the overall process variety. This restricts their utility in practical applications such as process standardization.

To address these shortcomings, this chapter introduces a novel measurement framework grounded in the diversity measure originally proposed by Weitzman [Wei92]. This framework enables a more nuanced quantification of structural process variety by explicitly incorporating pairwise structural disparities between traces. It also facilitates the ranking of trace variants based on their marginal contribution to overall variety.

However, the effectiveness of this approach is contingent upon the feasibility of constructing a valid hierarchical clustering of trace variants. In cases where no coherent taxonomy of process patterns exists, or where structural relationships are non-hierarchical, the applicability of the framework may be limited. Despite this constraint, the proposed approach establishes a theoretically grounded and practically relevant foundation for measuring structural process variety, while also high-

lighting the methodological boundaries that must be considered in its application.

4 Structural Trace Similarity

The selection of a suitable similarity measure for the pairwise comparison of process traces is not only crucial for the outcome of the introduced framework in the previous chapter 3.3, but also for various other process mining tasks. This includes trace clustering [SGvdA09, BCC19, ZRSH20], event abstraction [TSHvdA18, ZSDF+21], event log sampling [KNGW21], and conformance checking [LvdABP21]. Despite the extensive range of trace similarity measures available in the literature (for a comprehensive overview, see Back and Simonsen [BS23]), a formal comparison of these measures is still lacking. Consequently, interpreting the values derived from existing similarity measures is challenging, and selecting an appropriate measure for a specific process mining task remains uncertain.

One contributing factor to the limitations of existing similarity measures in process mining is that many of these techniques, such as edit distance and sequence alignment [BS23], originate from other research domains, including bioinformatics and computer science. These methods were originally developed for tasks such as comparing binary code [Lev66] or aligning DNA sequences [NW70], where the structural semantics differ significantly from those in process structure analysis. A major shortcoming of applying these measures to process mining is their inability to capture higher-level process patterns (cf. Sect. 2.1.1), such as loops, skips, and choices. These patterns are fundamental to accurately representing and comparing process structure [vdA22], and their omission can lead to oversimplified or

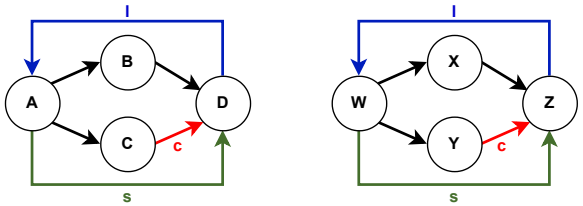


Figure 4.1: Graph representations for the traces $\sigma_1 = \langle A, B, D, A, C, D, A, D \rangle$ (left) and $\sigma_2 = \langle W, X, Z, W, Y, Z, W, Z \rangle$ (right), which are discovered based on the directly follows relations between the activities. The colored edges indicate three different process patterns: loop (l), choice (c), and skip (s).

misleading similarity assessments. For example, consider the traces $\sigma_1 = \langle A, B, D, A, C, D, A, D \rangle$ and $\sigma_2 = \langle W, X, Z, W, Y, Z, W, Z \rangle$. Traditional similarity measures based on edit distance (cf. Sect. 2.1.2) would assess these traces as entirely dissimilar due to the absence of shared activities. However, a closer examination of the ordering and repetition of activities reveals underlying structural similarities. As illustrated in Fig. 4.1, both traces exhibit identical process patterns, i.e., two loops, a choice, and a skip, each comprising subsequences of equal length.

Similarity measures commonly employed in process analysis, such as those reviewed by Back and Simonsen [BS23], are unable to detect the similarity between traces σ_1 and σ_2, as they rely on exact matching of activities to identify structural similarities. Consequently, these methods fail to recognize structural equivalence when traces exhibit similar patterns but use different activity labels. Beyond this label dependency, two additional limitations are noteworthy: first, these measures typically capture only a narrow subset of structural features, while overlooking complex process patterns. Second, they often involve high computational overhead, making them unsuitable for large-scale event logs or online settings where timely evaluation is critical [Bur22].

To address these limitations, a graph-based approach is introduced, capable of efficiently extracting high-level process patterns, including loops, skips, and choices. This extraction process involves transforming the traces into a graph representation using an adjacency list [ZWW+10]. The subsequent application of graph algorithms [CLRS22] facilitates the identification of pattern types within a trace, as well as the determination of their lengths, measured by the number of activities in the corresponding subsequences. This, in turn, enables a more comprehensive structural comparison between the traces.

This chapter is organized into five main sections. Sect. 5.1 reviews the relevant literature and identifies existing similarity measures that have been applied in process analysis. Sect. 4.2 introduces a set of formal properties that reflect various structural trace characteristics that a similarity measure should consider. These defined properties are subsequently utilized for a comprehensive comparison of the identified measures from the literature review. Building on the identified shortcomings from the formal comparison, Sect. 4.3 presents the graph-based measurement approach, which addresses these shortcomings. Based on this graph-based approach, six alternative similarity measures are introduced, each considering different subsets of structural features. Sect. 4.4 demonstrates the superiority of these measures over existing similarity measures through empirical evaluation. The evaluation is conducted using both syntactic and real-world data. The results of the evaluation are discussed in Sect. 4.5. Finally, Sect. 4.6 provides a short summary of this chapter.

4.1 Related Work

The evaluation of structural similarities and differences between traces can be considered from different research perspectives. Two particularly relevant research areas, which are concerned with the measure-

ment of distances based on structural process properties are *comparative trace analysis* (Sect. 4.1.1) and *comparative graph analysis* (Sect. 4.1.2). Furthermore, the research on process discovery algorithms is generally concerned with the identification and extraction of structural process features (Sect. 4.1.3).

4.1.1 Comparative Trace Analysis

Both ***syntactic*** and ***feature-based similarity measures*** (as described in Sect. 2.1.2) have been extensively utilized for the analysis of process traces [DDVD+11, BL12, SGvdA09, ZRSH20, BS23] [1].

Syntactic Similarity Measures

One of the most extensively used syntactic similarity measures in process analysis is the ***edit distance***, with its two variants: ***Levenshtein distance*** and ***normalized Levenshtein distance*** [JCBvdA09c, JCBvdA09a, DDVD+11, BL12, TSHvdA18, LvdABP21]. An additional variant of edit distance, proposed by [JCBvdA09c], adjusts editing costs based on the context in which they occur. These cost values are calculated based on the relative frequency of activity pairs within the traces, without explicitly considering structural features.

An alternative syntactic similarity measure is ***sequence alignment***, which is commonly applied to investigate the conformance between process models and event logs [JCBvdA10, KNGW21, TLRDM21, BS23]. Similar to edit distance, sequence alignment is calculated by optimizing a given scoring function, taking into account the operations necessary to align one trace with another.

[1] Back and Simonsen [BS23] make the distinction between *syntactic similarity metrics* and *vector-space embedding*. However, this classification does not include measures based on set comparison (cf. Sect. 2.1.2), such as Jaccard distance.

Feature-based Similarity Measures

A common approach for calculating feature-based similarities involves transforming traces into sets of ***n-grams*** (as described in Sect. 2.1.2) [SGvdA09, JCBvdA09c, KWW11, TSHvdA18, KNGW21, TLRDM21, TOJC23]. However, the length of n varies across different studies. Longer n-grams ($n > 2$) can capture more complex structural properties, such as loops of length > 2, but may fail to capture structural properties in shorter n-grams ($n \leq 2$). One solution is to aggregate similarity measures by considering n-grams of different lengths. While Delias et al. [DDG+15] propose combining 1- and 2-grams, Back and Simonsen [BDS19] suggest combining all possible substrings of a trace, from length one to length $n = min(|\sigma_i|, |\sigma_j|)$. However, this approach is computationally expensive and fails to differentiate between subsequences that correspond to distinct process patterns, such as sequences, loops, skips, or choices.

To derive a similarity value based on n-grams, existing studies either apply ***vector-space embedding***, including ***Euclidean*** and ***cosine distance*** [SGvdA09, JCBvdA09c, ZRSH20], or ***set comparison***, based on ***Jaccard distance*** [SGvdA09, KWW11, KNGW21, BLS22]. However, none of the identified studies provides a direct comparison between the two feature-based measurement approaches.

An alternative to trace comparison based on n-grams is the comparison based on ***sequential patterns*** [JCBvdA09a], i.e., ordered sequences of activities that occur frequently within a trace. Sequential pattern mining algorithms are commonly applied in various domains, such as bioinformatics, e-learning, market basket analysis, and click-stream analysis of webpages [FVLK+17]. Although numerous sequential pattern mining algorithms exist in the literature (for an overview, see [FVLK+17]), they are not commonly applied in the context of process

analysis. The comparative study by [BS23] demonstrated that similarity measures based on sequential patterns do not perform as well as other measures. One reason for this could be that the algorithms detect very detailed sequential patterns specific to single traces, failing to capture structural features common among multiple traces. Therefore, the analysis is restricted to similarity measures using 1-grams and 2-grams.

Beyond structural trace characteristics, additional features can be extracted from a trace and used for comparison. One such feature is the relative frequency of n-grams within a trace, which can be captured by ***eventually-follows relationships*** between activities. Delias et al. [DDG+15] propose an aggregated similarity measure based on cosine distance, considering 1-grams and eventually-follows relations. According to the study by Back and Simonsen [BS23], this measure yields particularly high performance for various event logs.

Moreover, alternative process perspectives have been considered for the comparison of traces, such as performance and resource attributes, which relate to the events of a trace [PRM16, TSHvdA18, KNGW21, TLRDM21]. However, for the study of structural similarity, only structural trace properties, commonly referred to as control-flow perspective, are considered.

4.1.2 Comparative Graph Analysis

Process model similarity measures, employed for comparing process models, mainly evaluate structural elements that are explicitly encoded in the control-flow representation of the model, such as branching constructs (for comprehensive overviews, see Dijkman et al. [DDVD+11], Becker and Laue [BL12], and Schoknecht et al. [STF+17]). These measures primarily focus on the comparison of process model elements, including node types and edges. The results of these comparisons are

contingent upon the labeling of nodes and edges, as well as the specific modeling language employed. However, there also exist some alternative measures, which consider the pairwise comparison of sets of traces, which are derived from the execution or simulation of the process models. These measures bear resemblance to those identified for trace comparison in the previous Sect. 4.1.1), with the distinction that they incorporate an additional step to aggregate the structural properties of the entire set. Nevertheless, these measures do not encompass process patterns such as loops, skips, and choices in the comparison.

An alternative group of measures for the pairwise comparison of process graphs are ***graph similarity measures***. These measures focus solely on the comparison of graph structures, disregarding specific process model elements and patterns (for a comprehensive overview, see Emmert-Streib et al. [ESDS16]). Hence, these measures consider some of the structural properties of a graph, e.g., by calculating the *edit distance* that indicates the number of changes that are required to transform one graph into another [GXTL10]. However, similarly to the process model similarity measures, graph similarity measures also fail to account for process-specific patterns such as loops, skips, and choices in the comparison.

In summary, existing similarity measures developed for process model and graph comparison exhibit several limitations when applied to the comparison of individual traces. These limitations include a reliance on specific modeling languages, a focus on comparing sets of traces rather than individual instances, and the inability to capture recurring process patterns such as loops, choices, and skips. To overcome these shortcomings, a graph-based similarity approach tailored to trace-level comparison is introduced in Sect. 4.3. This approach involves constructing a trace graph for each individual trace, which requires the

application of a process discovery algorithm to extract the underlying process structure.

4.1.3 Process Discovery Algorithms

Process discovery can be understood as the task of extracting structural features from a set of traces, typically with the goal of constructing a process model that generalizes the observed process structure. A variety of process discovery algorithms have been developed to identify both sequential patterns [DGP16, LvdA14] and non-sequential patterns [TSHvdA16], which represent recurring structural properties within an event log.

However, these algorithms generally operate under the assumption that the underlying process structure can be abstracted from multiple distinct traces, each representing a separate execution of the same process. To produce interpretable models, this often requires a high level of abstraction, which may obscure fine-grained structural details. Moreover, many discovery techniques prioritize frequently recurring patterns, potentially ignoring infrequent but meaningful structural patterns [LvdA14, TSHvdA16]. As a result, the complete set of structural patterns present in a single trace is not guaranteed to be captured.

Some algorithms are further limited by their focus on specific types of patterns. For instance, instance graphs [DGP16] and frequent episode mining [LvdA14] primarily target the detection of parallelism. However, identifying parallel execution from a single trace is inherently challenging, as the number of possible activity orderings increases exponentially with the number of concurrently executable activities. This complexity necessitates a high degree of abstraction, which can lead to the loss of important structural nuances.

Consequently, fine-grained structural properties may be overlooked. For example, given the trace $\sigma_1 = \langle A, B, C, D, A, C, B, D \rangle$, a process discovery algorithm such as the Local Process Model miner [TSHvdA16] might infer that activities B and C are executed in parallel. This would suggest structural equivalence with the trace $\sigma_2 = \langle A, C, B, D, A, B, C, D \rangle$. However, such an interpretation can be misleading in contexts like user behavior analysis [AAZB+21], where the specific order of actions, such as mouse clicks or visual fixations, may reflect distinct cognitive strategies. For instance, the sequence of interactions may indicate whether a user is engaging in exploratory versus goal-directed behavior [AAZB+21], which in turn can be linked to different cognitive processes [SAAW24, FAAS+24, BSP19]. These behavioral differences may warrant distinct interpretations and tailored forms of user support.

Finally, while various structural patterns, such as parallelism [DGP16, LvdA14, TSHvdA16], and declarative relationships [BLS22], can theoretically be derived from traces through further abstraction, the present study does not aim to evaluate the relevance of all such patterns for trace comparison. Instead, the focus is restricted to a subset of fundamental structural features that can be extracted with minimal abstraction, ensuring interpretability and applicability in fine-grained structural analysis.

To allow for a more precise comparison between the identified similarity measures, the following section introduces several formal properties, which help to distinguish between the structural trace features that are reflected by a given measure.

4.2 A Formal Comparison of Similarity Measures

For the formal comparison of the identified similarity measures, 10 distinct measurement properties are considered. This includes the four

metric properties already introduced in Sect. 2.1.2: P1 *non-negativity*, P2 *symmetry*, P3 *identity of indiscernibles*, and P4 *triangle-inequality*. Furthermore, Sect. 4.2.1 defines three additional properties, which describe the measurement behavior with respect to the structural properties of the traces. Finally, Sect. 4.2.2 defines three generic properties for the measurement of similarities between traces, which can be considered desirable in the context of comparative process analysis.

4.2.1 Properties Related to Trace Structure

The comparative analysis conducted by Back and Simonsen [BS23] demonstrates that trace similarity measures exhibit varying performance contingent upon the characteristics of an event log, including the number of activities, the trace lengths, and the number of trace classes. Although the comparative analysis by Back and Simonsen offers some insights into the applicability of similarity measures, it fails to account for how these measures represent specific trace characteristics, such as structural features. To address this limitation, three properties are identified, which allow to determine the ability of a similarity measure to reflect structural trace properties. These properties are based on the assumption that a similarity measure should increase monotonically with the number of structural differences between traces. Three types of structural differences are considered: (1) activities, (2) directly follows relations, and (3) high-level structural features (loops, skips, and choices). These features offer a detailed characterization of a trace's structure and are widely acknowledged in the literature as theoretically significant for structural process analysis [DDVD+11, BL12, TLRDM21, vdA22].

The first structural property pertains to the differences among activities. A similarity measure should exhibit a strictly increasing relationship with respect to differences in the number of distinct activities. This

principle is based on the intuitive notion that greater dissimilarity between traces corresponds to more distinct sets of activities. To compare the dissimilarity between two sets, their symmetric difference is considered: $F_i \triangle F_j = (F_i \cup F_j \setminus F_i \cap F_j)$. The set of activities within a trace can be derived using the feature-extraction function based on 1-grams (cf. Sect. 2.1.2), i.e., $h_1(\sigma_i) = F_{i,1}$. A similarity measure should be increasing with an increase in the number of distinct activities between two traces, i.e.:

Property P6. $d(F_{i,1}, F_{j,1}) < d(F_{i,1}, F_{k,1})$ *for all* $|F_{i,1} \triangle F_{j,1}| < |F_{i,1} \triangle F_{k,1}|$.

When comparing the traces σ_0, σ_3 and σ_4 shown in Fig. 4.2, according to property P6 one would expect $d(\sigma_0, \sigma_3) < d(\sigma_0, \sigma_4)$ since the sets of activities contained in σ_0 and σ_4 are more distinct from one another than in σ_0 and σ_3.

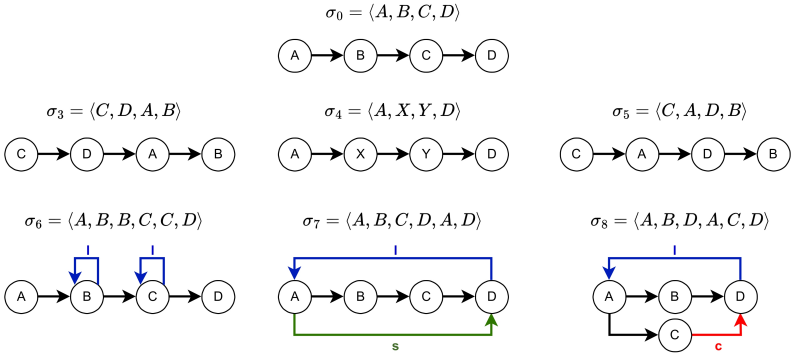


Figure 4.2: Different traces and their representation as trace graph.

The second structural property pertains to the differences among directly follows relations. A similarity measure should exhibit a strictly increasing relationship with respect to the number of distinct directly follows relations. This principle is based on the intuitive notion that

greater dissimilarity between traces corresponds to more distinct orderings of activities in terms of their directly follows relations. The set of directly follows relations within a trace can be derived using the feature-extraction function based on 2-grams (cf. Sect. 2.1.2), i.e., $h_2(\sigma_i) = F_{i,2}$. Consequently, a similarity measure should be increasing with an increase in the number of distinct directly follows relations between two traces, i.e.:

Property P7. *$d(F_{i,2}, F_{j,2}) < d(F_{i,2}, F_{k,2})$ for all $|F_{i,2} \triangle F_{j,2}| < |F_{i,2} \triangle F_{k,2}|$.*

When comparing the traces σ_0, σ_3 and σ_5 shown in Fig. 4.2, according to Property P7 one would expect $d(\sigma_0, \sigma_3) < d(\sigma_0, \sigma_5)$, since the sets of directly follows relations contained in σ_0 and σ_5 are more distinct from one another than in σ_0 and σ_3.

Based on n-grams it is possible to capture structural properties of a trace σ_i, such as the activities in σ_i (based on 1-gram), or the directly follows relationships in σ_i (based on 2-gram). However, identifying high-level structural features, such as loops, skips, and choices, requires a mapping of a trace to a process graph based on some discovery function $f_{dis} : \mathcal{B}(\mathcal{A}^*) \rightarrow \mathcal{G}$ (as defined in Sect. 2.1). To reflect the structural properties of a trace, the discovered process graph, which will be termed *trace graph* in the following, must respect the order of the activities in the trace.

Definition 16 (Trace Graph) *Let $A = F_{i,1}$ and $D = F_{i,2}$ denote the activities and the directly follows relations that can be seen in a given trace $\sigma_i = \langle a_1, a_2, ..., a_n \rangle$. A trace graph is defined as a process graph $G(A, D)$, where A and D visualize the activities and directly follows relations of σ_i, such that $A = \{a_1, a_2, ..., a_n\}$ and $D = \{(a_1, a_2), ..., (a_{n-1}, a_n)\}$.*

Given a discovered trace graph $G(A, D) \in \mathcal{G}$, it is thus possible to derive additional graph features, which reflect structural properties of the

original trace. This can be achieved by distinguishing between different edge types within the graph.

Definition 17 (edge-type) *Let F_e be a set of edge features. Edge-type is a feature-extraction function $h_e : \mathcal{G} \rightarrow \mathcal{P}(F_e)$, which defines for a given trace graph $G(A, D)$ a set of edge features F_e, where each feature is a tuple $f_e = (d, w, l) \in F_e$ describing an edge $d \in D$, the edge type $w \in \{sequence, loop, skip, choice\}$ and a length $l \in \mathbb{N}$ associated with the edge type.*

The two trace graphs in Fig. 4.1 encompass all four defined edge types. These edge types correspond to four distinct structural characteristics frequently observed in process traces: the sequential order of activities (sequence), the repetition of activities (loop), the omission of activities (skip), and the selection of alternative activities (choice). While the lengths of loops, skips, and choices can vary (with a minimum length of one), the length of a sequence is always one.

Beyond the set of activities and their relative ordering, a similarity measure should also account for process patterns present in the traces based on edge types. The set of edge types within a trace can be derived using the feature-extraction function $h_e(f_{dis}(\sigma_i)) = F_{i,e}$. A similarity measure should be increasing with an increase in the number of distinct edge types between two traces, i.e.:

Property P8. $d(F_{i,e}, F_{j,e}) < d(F_{i,e}, F_{k,e})$ for all $|F_{i,e} \triangle F_{j,e}| < |F_{i,e} \triangle F_{k,e}|$.

When comparing the traces σ_0, σ_3, σ_6, σ_7 and σ_8 shown in Fig. 4.2, according to Property P8 one would expect $d(\sigma_0, \sigma_3) < d(\sigma_0, \sigma_6)$ due to the loop in σ_6. Similarly, one would expect $d(\sigma_0, \sigma_3) < d(\sigma_0, \sigma_7)$ due to the loop and skip in σ_7, and $d(\sigma_0, \sigma_3) < d(\sigma_0, \sigma_8)$ due to the loop and choice in σ_8.

4.2.2 Generic Properties

In addition to the structure-related properties P6-P8, three further generic properties related to the similarity measurement of traces are defined. A similarity measure should also account for the overall number of features present within the traces. For instance, the similarity between $\sigma_0 = \langle A, B, C, D \rangle$ and $\sigma_9 = \langle A, B, C, D, X \rangle$ should be deemed greater than the similarity between $\sigma_{10} = \langle A \rangle$ and $\sigma_{11} = \langle A, X \rangle$, despite both pairs of traces differing by only a single activity. Consequently, the contribution of a new feature to the dissimilarity between two traces should diminish as the overall set of features in both traces increases. This property is referred to as *submodularity* and has also been deemed relevant for the measurement of diversity in general [NP09].

Property P9. *For all* $(F_i \cup F_j) \subseteq (F_k \cup F_l)$ *and* $f \notin F_k, F_l$ *we have that* $d(F_i, F_j \cup f) - d(F_i, F_j) \geq d(F_k, F_l \cup f) - d(F_k, F_l)$.

This property only considers the overall set of features contained in the traces, without accounting for the ratio between similar and dissimilar features. However, following Becker et al. [BL12], a comparative process analysis based on some similarity measure should incorporate both the commonalities and the differences between the traces.

Property P10. *A similarity measure should consider both commonalities* $F_i \cap F_j$ *and differences* $F_i \triangle F_j$ *between two traces.*

Finally, the computational costs of the similarity measures are evaluated to ensure their scalability for large-scale event logs and applicability in real-time or online processing environments. This consideration again follows from Becker et al. [BL12], who deem computational efficiency as a relevant property for comparing similarity measures in the context of process analysis. Computational efficiency is often assessed

in terms of achieving linear time complexity, as defined in algorithmic analysis [CLRS22].

Property P11. *The similarity measure can be calculated efficiently.*

Table 4.1 shows an overview of all considered properties for the formal comparison of similarity measures, including the metric properties defined in Sect. 2.1.2, structure-related properties, and generic properties.

Metric	**P1**	Non-negativity
	P2	Symmetry
	P3	Identity of indiscernibles
	P4	Triangle-inequality
Structure	**P6**	Monotonicity with an increase in the number of distinct activities
	P7	Monotonicity with an increase in the number of distinct directly follows relations
	P8	Monotonicity with an increase in the number of distinct edge types
Generic	**P9**	Submodularity
	P10	Consideration of commonalities and differences
	P11	Computational efficiency

Table 4.1: Overview of the desirable properties for similarity measures.

4.2.3 Formal Comparison

The properties outlined in Table 4.1 serve as the basis for comparing the similarity measures listed in Table 4.2. The selected measures for the comparison are commonly used in process analysis (cf. Sect.4.1.1) and focus on the comparison of structural trace features. Additionally, these measures have demonstrated a high performance in the comparative study by Back and Simonsen [BS23].

For the evaluation of properties related to trace structure P6-P8, the calculated similarities between σ_0 and $\sigma_3 - \sigma_8$ are considered (cf. Fig. 4.1). When comparing σ_0 with $\sigma_4 - \sigma_8$, the set of distinct structural features is consistently larger than the set of distinct structural features that can be derived from the comparison between σ_0 and σ_3.

Type	Measure	Label
Syntactic	Levenshtein distance	LD
	Normalized Levenshtein distance	N-LD
Feature-based Vector-space embedding	Euclidean distance based on MR	EMR
	cosine distance based on MR	CMR
	Jaccard distance based on MR	JMR
	Eventually-Follows	EF
	Euclidean distance based on 1-gram	E1
	Euclidean distance based on 2-gram	E2
	Euclidean distance based on 3-gram	E3
	cosine distance based on 1-gram	C1
	cosine distance based on 2-gram	C2
	cosine distance based on 3-gram	C3
Feature-based Set comparison	Jaccard distance based on 1-gram	J1
	Jaccard distance based on 2-gram	J2
	Jaccard distance based on 3-gram	J3

Table 4.2: Selection of similarity measures.

Thus, for a similarity measure, which is strictly increasing with respect to the number of distinct activities P6, directly follows relations P7, and edge types P8 it should be true that $d(\sigma_0, \sigma_3) < (\sigma_0, \sigma_{4-8})$. The derived values for the respective similarity measures in Table 4.2 are presented in Table 4.3.

Table 4.4 describes the analyzed similarity measures with an indication of their adherence to the considered metric properties P1-P4, structural properties P6-P8, and generic properties P9-P11. For the metric properties P1-P4, it is evident that the Levenshtein distance (LD) and Jaccard distance (J1 and J2) are the only measures conforming to all four metric properties [YB07, Kos19]. Conversely, the normalized Levenshtein distance (N-LD) and cosine distance measures (C1 and C2) fail to qualify as metrics due to their violation of the triangle inequality

	$s(\sigma_0, \sigma_3)$	$s(\sigma_0, \sigma_4)$	$s(\sigma_0, \sigma_5)$	$s(\sigma_0, \sigma_6)$	$s(\sigma_0, \sigma_7)$	$s(\sigma_0, \sigma_8)$
LD	4.000	2.000	4.000	2.000	2.000	2.000
N-LD	1.000	0.500	1.000	0.333	0.333	0.333
EMR	0.000	0.000	0.000	1.414	1.414	1.414
CMR	0.000	0.000	0.000	0.000	0.000	0.000
JMR	0.000	0.000	0.000	0.000	0.000	0.000
EF	0.179	0.150	0.299	0.169	0.072	0.113
E1	0.000	2.000	0.000	1.414	1.414	1.414
E2	2.449	2.449	3.162	1.414	1.414	2.000
E3	2.828	2.828	2.828	2.449	2.000	2.449
C1	0.000	0.500	0.000	0.051	0.051	0.051
C2	0.600	0.600	1.000	0.155	0.155	0.324
C3	1.000	1.000	1.000	0.592	0.388	0.592
J1	0.000	0.667	0.000	0.000	0.000	0.000
J2	0.750	0.750	1.000	0.286	0.286	0.500
J3	1.000	1.000	1.000	0.750	0.571	0.75

Table 4.3: Calculated similarity values for the pairwise comparison between the reference trace σ_0 and traces σ_3 through σ_8, with $s(\sigma_0, \sigma_3)$ serving as the baseline. According to the defined structural properties P6 to P8, specific expectations are associated with each comparison. The similarity value $s(\sigma_0, \sigma_4)$ is expected to reflect an increase in the number of activities, as defined by property P6. Similarly, $s(\sigma_0, \sigma_5)$ should capture an increase in the number of directly follows relations, in accordance with property P7. The comparisons $s(\sigma_0, \sigma_6)$ through $s(\sigma_0, \sigma_8)$ are intended to reflect an increase in the number of edge types, as specified by property P8. Any similarity value that is less than or equal to the baseline $s(\sigma_0, \sigma_3)$ indicates a violation of the corresponding property and is highlighted in gray to emphasize this deviation.

[YB07, HKP12]. Similarly, the Eventually-Follows measure (EF), derived from cosine distance, does not meet the metric criteria.

In terms of structure-related properties P6-P8, it is apparent that none of the measures fully comply with more than one of the properties. Specifically, only C1 and J1 accurately assess the dissimilarity in the

	Metric				Structure					Similarity
	P1	**P2**	**P3**	**P4**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**
LD	yes	yes	yes	yes	no	no	no	no	no	$\mathcal{O}(\lvert\sigma_1\rvert \cdot \lvert\sigma_2\rvert)$
N-LD	yes	yes	yes	no	no	no	no	no	no	$\mathcal{O}(\lvert\sigma_1\rvert \cdot \lvert\sigma_2\rvert)$
EMR	yes	yes	yes	yes	no	no	yes	no	yes	$\mathcal{O}(\lvert F_{i,MR} \cup F_{j,MR}\rvert)$
CMR	yes	yes	yes	no	no	no	no	no	no	$\mathcal{O}(\lvert F_{i,MR} \cup F_{j,MR}\rvert)$
JMR	yes	yes	yes	yes	no	no	no	yes	no	$\mathcal{O}(\lvert F_{i,MR} \cup F_{j,MR}\rvert)$
EF	yes	yes	yes	no	no	yes	no	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2}\rvert)$
E1	yes	yes	yes	yes	yes	no	yes	no	yes	$\mathcal{O}(\lvert F_{i,1} \cup F_{j,1}\rvert)$
E2	yes	yes	yes	yes	no	yes	no	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2}\rvert)$
E3	yes	yes	yes	yes	no	no	no	no	yes	$\mathcal{O}(\lvert F_{i,3} \cup F_{j,3}\rvert)$
C1	yes	yes	yes	no	yes	no	yes	no	yes	$\mathcal{O}(\lvert F_{i,1} \cup F_{j,1}\rvert)$
C2	yes	yes	yes	no	no	yes	no	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2}\rvert)$
C3	yes	yes	yes	no	no	no	no	no	yes	$\mathcal{O}(\lvert F_{i,3} \cup F_{j,3}\rvert)$
J1	yes	yes	yes	yes	yes	no	no	yes	yes	$\mathcal{O}(\lvert F_{i,1} \cup F_{j,1}\rvert)$
J2	yes	yes	yes	yes	no	yes	no	yes	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2}\rvert)$
J3	yes	yes	yes	yes	no	no	no	yes	yes	$\mathcal{O}(\lvert F_{i,3} \cup F_{j,3}\rvert)$

Table 4.4: Comparison between similarity measures based on the identified desirable properties P1-P4 and P6-P11.

number of activities P6. Additionally, C2, J2, and EF effectively evaluate the dissimilarity in directly follows relations P7.

Concerning the general similarity properties P9-P11, J1 and J2 are the only measures that satisfy submodularity P9 (a proof is provided by Kosub [Kos19]). Furthermore, all similarity measures, except LD and N-LD, account for both similarities and dissimilarities between traces P10. Lastly, all similarity measures can be computed in linear time P11, contingent upon the length of the traces or the respective sets of features.

In summary, only the Jaccard distance measures (J1 and J2) satisfy all metric properties (P1-P4), satisfy submodularity (P9) and recognize both dissimilarities and similarities between traces (P10). What is even more important to recognize, with regard to the comparative analysis

of trace structures, is that none of the considered similarity measures adequately evaluate structural dissimilarities between traces (P6-P8). Structural dissimilarity is only partially evaluated correctly (by C1 and J1) for activities (P6) and only partially evaluated correctly (by C2, J2, and EF) for directly follows relations (P7). The subsequent section introduces a graph-based approach, which addresses this issue by integrating additional structural features into the similarity evaluation.

4.3 Graph-based Similarity Measures

To incorporate additional structural features in the trace comparison, Sect. 4.3.1 introduces a graph-based approach, which allows to identify different process patterns that are contained in a trace. Following this approach, six novel similarity measures are introduced, each emphasizing distinct structural trace characteristics. Subsequently, Sect. 4.3.2 provides a formal comparative analysis of these measures, which demonstrate their superiority over the analyzed similarity measures in Sect. 4.2.3.

4.3.1 Enhanced Similarity Measurement with Graph Features

The graph-based approach introduced in this section involves two main steps. First, the trace is converted into a trace graph $G(A, D)$ represented as an adjacency list. In an adjacency list, each node $a \in A$ is associated with a list detailing all adjacent nodes connected by an edge $d \in D$ [CLRS22]. This transformation introduces a level of abstraction. For instance, the adjacency list does not consider the relative frequencies of the directly follows relations in a trace. The abstraction, however, facilitates a direct and computationally efficient comparison of trace structures [ZWW+10].

Second, for the identification of the graph's edge types (i.e., sequence, loop, skip, and choice) and their respective lengths, three different graph algorithms are employed: depth-first search (DFS), breadth-first search (BFS), and lowest-common ancestor (LCA). Overall, the algorithmic approach involves three main steps.

Step 1: Identifying Edge Types

The DFS allows identifying the type of edges for all edges in a graph [CLRS22]. Beginning with the edge that connects the start node (according to the start activity defined by the trace) with its first subsequent node (according to the first directly follows relation defined by the trace), the ***DFS algorithm*** iteratively assigns a type to all edges in a graph. In each step, the algorithm evaluates whether an edge has been previously visited and which other edges have already been traversed. Accordingly, the edge types are assigned as described by Algorithm 1. These assigned edge types can be visualized within a trace graph using different colors, as shown in Fig. 4.1.

Since the subsequences that manifest a process pattern, such as a loop, a skip, or a choice (as described in Sect. 2.1.1) can be of different lengths, it is reasonable to also add this length as a description of an edge, to further detail the edge type. For instance, the trace $\sigma_1 = \langle A, B, D, A, C, D, A, D \rangle$ illustrated in Fig. 4.1 encompasses the edge (D, A) of the type loop with the length one, the edge (A, D) of type skip with the length one, and the edge (C, D) of the type choice with the length one. For each edge the lengths is determined by the minimum length of the subsequences that manifest the type of an edge. Edges of type "sequence" are thereby considered to always be of length one, since they indicate directly follows relationships between activities.

Algorithm 1 DFS to detect graph-features including length

Input: $G(A, D)$ in the form of adjacency list $AdjList$
Output: List of tuples (d, w, l) for all edges $d \in D$ with the respective edge type $w \in$ {sequence, loop, skip, choice} and length $l \in \mathbb{N}$

1: main() function
2: $i \leftarrow 0$ ▷ initialize counter
3: $numList(|v|) \leftarrow 0$ ▷ initialize list with counters of node visits
4: $recList(|v|) \leftarrow 0$ ▷ initialize list with nodes currently on recursion stack
5: **for** all $v \in AdjList$ **do**
6: **if** $numList(v) = 0$ **then** dfd(v) ▷ apply recursive DFS function
7:
8: dfd() function
9: $i \leftarrow i + 1$, $numList(v) \leftarrow i$, $recList(v) \leftarrow 1$
10: **for** all $w \in adj(v)$ **do** ▷ $adj(v)$ is the list for $v \in AdjList$
11: **if** num(w)=0 **then** dfd(w) ▷ $\langle v, w \rangle$ is a sequence edge
12: **else if** $numList(w) > numList(v)$ **then** ▷ $\langle v, w \rangle$ is a skip edge
13: **call** BFd($start = w$, $end = v$) ▷ derive length based on Algo. 2
14: **return** ($type = forward$, length, $\langle v, w \rangle$)
15: **else if** $numList(w) < numList(v)$ **then** ▷ $\langle v, w \rangle$ is a choice edge
16: **call** LCA(w,v) ▷ derive length based on Algo. 3
17: **return** ($type = cross$, length, $\langle v, w \rangle$)
18: **else** ▷ $\langle v, w \rangle$ is a loop edge
19: **call** BFd($start = w$, $end = v$) ▷ derive length based on Algo. 2
20: **return** ($type = back$, length, $\langle v, w \rangle$)
21: $recList(v) \leftarrow 0$

Algorithm 2 BFS

Input: $G(A, D)$ in the form of adjacency list $AdjList$, start node a, end node b
Output: Minimum number of steps from a to b, list of all visited nodes

```
1: bfd() function
2: queueList ← (a, steps = 0)                          ▷ initialize queue
3: visitedList ← 0                                     ▷ initialize list of visited nodes
4: while queueList not empty do
5:     currentnode, steps ← last entry queueList
6:     if currentnode not in visitedList then
7:     visitedList ← currentnode
8:     if currentnode = b
9:         return steps, visitedList
10:    for all neighboringnode of currentnode ∈ AdjList do
11:        queueList ← (neighboringnode, steps + 1)
12:        if neighboringnode not in visitedList then
13:            visitedList ← (neighboringnode, steps)
```

Algorithm 3 LCA

Input: $G(A, D)$ in the form of adjacency list $AdjList$, node a, node b
Output: Lowest-common ancestor of a and b, and the min number of steps

```
1: remove edge ⟨a, b⟩ from AdjList
2: AdjList' ← invert(AdjList)
3: LCA() function
4: v ← startnode(AdjList')
5: l1 = BFd(v, a)    ▷ derive list of visited nodes with min. steps based on Algo.2
6: l2 = BFd(v, b)    ▷ derive list of visited nodes with min. steps based on Algo.2
7: x ← firstCommonnode(l1,l2)
8: return min number of steps between a and x, b and x
```

Step 2: Identifying the Length Associated with an Edge Type

To compute the length associated with an edge for a given trace, the DFS is combined with the two graph algorithms BFS and LCA. The ***BFS algorithm*** [CLRS22] derives the minimum number of steps required to traverse from one node $a \in A$ to another node $b \in A$ following the edges in a graph. The algorithm starts with a given node $a \in A$ and explores the neighboring nodes at the current depth before progressing to the nodes at the subsequent depth level. As shown by Algorithm 2 a queue data structure can be utilized to monitor the nodes that remain to be explored. This procedure continues until the node $b \in A$ is located. The final depth signifies the minimum number of steps to transition from a to b. Hence, to calculate the length of the edge (B, A) in σ_1, which is of type loop, the BFS is applied to compute the minimum number of steps required to transition from A to B. Similarly, to calculate the length of the edge (A, D) in σ_1, which is of type skip, the BFS is applied to compute the minimum number of steps required to transition from B to A.

To derive the length for an edge of type choice, requires first the identification of the initial activity that precedes the alternative execution paths, considered as choice (cf. Sect. 2.1.1. For example, the two alternative execution paths in $\sigma_8 = \langle A, B, D, A, C, D \rangle$ (see Fig. 4.1) are preceded by the activity A. One way to identify the preceding activity is to apply the ***LCA algorithm*** [CLRS22]. As shown by Algorithm 3, the algorithm initially requires the inversion of the trace graph, such that all start nodes become the end nodes of the graph. Subsequently, the LCA algorithm identifies the lowest common ancestor of two given nodes $a, b \in A$ by iteratively traversing through the graph. At each step, the neighborhood of a is compared with the neighborhood of b, until a common descendant for both activities is found.

Step 3: Aggregating the Features

Once, the edge types and their lengths are derived based on the described algorithms, this information can be incorporated into the measurement of trace similarity. To evaluate the impact of the derived graph-based features for the structural trace comparison, six different measures are proposed. As shown in Table 4.6 these measures are either calculated based on cosine distance (vector-space embedding) or Jaccard distance (set comparison). For each type of measure, different sets of features are considered: (1) graph-based features (i.e., edge type and length), (2) 1-grams and 2-grams, and (3) 1-grams, 2-grams, and graph-based features.

Graph-based cosine distance (Cg) is computed using the identified edge types, incorporating both their lengths and frequencies. In contrast, graph-based Jaccard distance (Jg) is calculated based on the set of directly follows relationships, each annotated with its corresponding edge type and length. Hence, both measures capture the frequency of the high-level process patterns within traces.

Edges of type sequence are excluded from the similarity computation, as their contribution is already implicitly reflected in the lengths of the loops, skips, and choices. Including the number of sequence edges would disproportionately bias the similarity scores toward trace length. However, to ensure that traces composed solely of sequence edges (i.e., lacking higher-level patterns) are still recognized as structurally similar, a synthetic structural feature labeled *sequ.* is added to all traces of length greater than one. An illustrative example of how Jg and Cg are calculated is provided in Table 4.5.

The aggregations (denoted by $Agg()$ in Table 4.6) are computed based on the sum of the individual measures with uniform weighting ap-

M.	Transformation	Calculation
Cg	$\vec{\sigma_0} = \begin{matrix}(sequ.)\\(loop,1)\end{matrix}\begin{bmatrix}1\\0\end{bmatrix}$ $\vec{\sigma_6} = \begin{matrix}(sequ.)\\(loop,1)\end{matrix}\begin{bmatrix}1\\2\end{bmatrix}$	$1 - \frac{\vec{\sigma_0}\cdot\vec{\sigma_6}}{\|\vec{\sigma_0}\|\|\vec{\sigma_6}\|} = 1 - \frac{1}{\sqrt{5}}$
Jg	$F_{\sigma_0} = \{sequ.\}$ $F_{\sigma_6} = \{sequ., (loop,1,\langle B,B\rangle), (loop,1,\langle C,C\rangle)\}$	$1 - \frac{F_{\sigma_0}\cap F_{\sigma_6}}{F_{\sigma_0}\cup F_{\sigma_6}} = \frac{2}{3}$

Table 4.5: Example for calculating the similarity between $\sigma_0 = \langle A,B,C,D\rangle$ and $\sigma_6 = \langle A,B,B,C,C,D\rangle$ based on the two measures Jg and Cg.

plied, e.g., $Agg(C1,C2) = C1+C2$. This aggregation approach has also been applied in related studies [KWW11, DDG+15, KNGW21, BLS22].

The following subsection provides an evaluation of the graph-based measures according to the measurement properties described in Sect. 4.2.1.

Type	Measure	Label
Vector-space emb.	Graph-based cosine distance	Cg
	Aggregated cosine distance	Agg(C1,C2)
	Aggreagted graph-based cosine distance	Agg(C1,C2,Cg)
Set comparison	Graph-based Jaccard distance	Jg
	Aggregated Jaccard distance	Agg(J1,J2)
	Aggreagted graph-based Jaccard distance	Agg(J1,J2,Jg)

Table 4.6: Set of proposed trace similarity measures.

4.3.2 Formal Comparison

The calculated similarity measures between σ_0 and $\sigma_3 - \sigma_8$ based on the six introduced measures in the previous section (shown in Table 4.6) are shown in Table 4.7.

Based on these calculated values, Table 4.8 gives a comprehensive overview of the similarity measures, indicating their adherence to the defined properties P1-P4 and P6-P11. Concerning the *metric properties*

(P1-P4), it can be observed that measures based on cosine distance generally do not satisfy the triangle inequality (P4) [HKP12]. Conversely, the measures based on Jaccard distance adhere to the triangle inequality (P4), even when they are aggregated [KWW11].

Concerning the *structure-related properties* P6-P8, it is shown that both Cg and Jg accurately reflect the dissimilarities between traces based on edge types. Moreover, *Agg(J1,J2,Jg)* is the sole measure that correctly captures the dissimilarity of all structural features, thus providing validation of the aggregation approach, i.e., the combination of individual measures. However, *Agg(C1,C2,Cg)*, does not correctly capture the dissimilarity of all structural features, since the cosine distance is biased by the frequency of the trace features.

Concerning the *generic similarity properties* (P9-P11) one can make the following observations. The Jaccard distance is the only measure type which can be considered a submodular function (P9). However, both measure types (Jaccard distance and cosine distance) account for similarities and dissimilarities between traces (P10). Finally, in terms of computational efficiency (P11), it should be noted that the additional calculations of graph-based features using Algorithm 1 do not increase the time complexity compared to the similarity measures from the literature, as discussed in Sect. 4.2. DFS and BFS both exhibit a linear time complexity of $\mathcal{O}(|A| + |D|)$. Similarly, LCA can be computed with linear time complexity $\mathcal{O}(h)$, where h denotes the length of the longest distinct path between two nodes in $G(A, D)$.

4.4 Empirical Evaluation

For the comparison of the measures discussed in Sect. 4.2 and the novel graph-based measures introduced in Sect. 4.3 this section provides a comprehensive empirical evaluation based on four synthetic event logs

	$s(\sigma_0, \sigma_3)$	$s(\sigma_0, \sigma_4)$	$s(\sigma_0, \sigma_5)$	$s(\sigma_0, \sigma_6)$	$s(\sigma_0, \sigma_7)$	$s(\sigma_0, \sigma_8)$
Cg	0.000	0.000	0.000	0.553	0.423	0.423
Agg(C1,C2)	0.600	1.100	1.000	0.206	0.206	0.375
Agg(C1,C2,Cg)	2.449	2.949	3.162	2.018	1.888	2.474
Jg	0.000	0.000	0.000	0.667	0.667	0.667
Agg(J1,J2)	0.750	1.417	1.000	0.286	0.286	0.500
Agg(J1,J2,Jg)	0.750	1.417	1.000	0.952	0.952	1.167

Table 4.7: Calculated similarity values for the pairwise comparison between the reference trace σ_0 and traces σ_3 through σ_8, with $s(\sigma_0, \sigma_3)$ serving as the baseline. According to the defined structural properties P6 to P8, specific expectations are associated with each comparison. The similarity value $s(\sigma_0, \sigma_4)$ is expected to reflect an increase in the number of activities, as defined by property P6. Similarly, $s(\sigma_0, \sigma_5)$ should capture an increase in the number of directly follows relations, in accordance with property P7. The comparisons $s(\sigma_0, \sigma_6)$ through $s(\sigma_0, \sigma_8)$ are intended to reflect an increase in the number of edge types, as specified by property P8. Any similarity value that is less than or equal to the baseline $s(\sigma_0, \sigma_3)$ indicates a violation of the corresponding property and is highlighted in gray to emphasize this deviation.

	Metric				**Structure**				**Similarity**	
	P1	**P2**	**P3**	**P4**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**
Cg	yes	yes	yes	no	no	no	yes	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$
Agg(C1,C2)	yes	yes	yes	no	yes	yes	no	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$
Agg(C1,C2,Cg)	yes	yes	yes	no	yes	yes	no	no	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$
Jg	yes	yes	yes	yes	no	no	yes	yes	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$
Agg(J1,J2)	yes	yes	yes	yes	yes	yes	no	yes	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$
Agg(J1,J2,Jg)	yes	yes	yes	yes	yes	yes	yes	yes	yes	$\mathcal{O}(\lvert F_{i,2} \cup F_{j,2} \rvert)$

Table 4.8: Comparison between similarity measures based on the identified desirable properties P1-P4 and P6-P11.

and four real-life event logs. Performance is assessed based on the accuracy with which each measure captures structural similarity between traces within the respective logs.

Each trace is associated with a unique class, where traces belonging to the same class are expected to exhibit higher structural similarity to one another than to traces from different classes. To quantify the

accuracy of the similarity assessments, three distinct evaluation metrics are applied.

The evaluation is designed to address two central research questions:

1. Do the proposed similarity measures outperform existing alternatives in terms of classification accuracy?
2. Do the introduced graph-based features improve the accuracy in trace similarity assessment?

The event logs and a Python implementation of all considered similarity measures can be accessed at:
https://github.com/promilab/StructuralProVar.git.

4.4.1 Evaluation Measures

The empirical evaluation is based on the following three evaluation measures introduced by Back and Simonsen [BS23]: *Nearest Neighbor*, *Precision@k*, and *Triplet*. The evaluation measures are well-established in the literature on similarity measurement [BHS13, KB19]. Additionally, Back and Simonsen [BS23] provide some validation of these measures through an extensive comparison of 18 trace similarity measures. A detailed description of how the measures are calculated can be found in the Appendix B.

In general, the evaluation measures aim to compare the distance values of the in-class data points, i.e., data points with the same class label, with the distance values of the out-of-class data points, i.e., data points with distinct class labels. A similarity measure is positively evaluated if the distance values between data points within the same class are high, and the distance values between data points from different classes are low.

Accordingly, *Nearest Neighbor* evaluates the performance of a similarity measure by determining the ratio of data points whose nearest neigh-

bor, i.e., the closest data point based on a given similarity measure, belongs to the same class versus those whose nearest neighbor belongs to a different class.

Precision@k extends this evaluation by calculating the ratio of in-class to out-of-class data points among the k-nearest data points. In their study, Back and Simonsen [BS23] determined the parameter k to be 10.

Triplet offers an even more granular evaluation of distances by considering, for each data point, all distances between in-class and out-of-class data points. Given an anchor data point x, the measure calculates, for each in-class data point y, the ratio of out-of-class data points z with $d(x, y) < d(x, z)$ to those with $d(x, y) > d(x, z)$.

A general concern regarding the introduced evaluation measures is their computational complexity, especially when it comes to the calculation of *Triplet*. Back and Simonsen [BS23] address this issue by restricting the size of the selected data sets for the evaluation. Similarly, the size of selected data sets for the empirical evaluation in this section is restricted to a maximum number of 350 traces.

4.4.2 Datasets

The empirical evaluation of the similarity measures introduced in Sect. 4.2 and Sect. 4.3 is based on seven different event logs. As shown in Table 4.9, they include three synthetic event logs derived from simulation (SynL1, SynL2, SynL3, SynL4) and four real-world event logs derived from some information system (WABO, PO, EyeT1, EyeT2).

The set of event logs was selected based on two primary criteria. First, the evaluation measures employed in this study require that each trace is associated with a distinct label, enabling its unambiguous assignment to a specific class. In this context, a class refers to a category or grouping of traces that exhibit similar structural characteristics. Sec-

	Activity	Trace	Min	Max	Class	T./C.	Class Type	Origin
SynL1	16	300	8	194	3	100	edge type	Simulation
SynL2	16	300	8	202	6	50	edge type & length	Simulation
SynL3	16	300	8	179	6	50	edge type & activity	Simulation
SynL4	16	320	8	202	4	80	edge type & activity	Simulation
WABO	465	350	1	110	5	70	process type	Administration
PO	42	300	8	324	3	100	process type	Administration
EyeT1	1194	349	27	1152	8	43-44	task type	Eye tracking
EyeT2	1194	349	27	1152	2	174-175	task complexity	Eye tracking

Table 4.9: Overview of the event logs including the number of activities, number of traces, the minimum and maximum length of traces, number of classes, number of traces per class, class type, and their origin.

ondly, classes must be distinguishable based on the structural characteristics of the traces within them. This entails distinct structural features related to activities, directly follows relations, and edge types, as defined by properties P6-P8 in Sect.4.2. The distinction between classes based on edge types P8 is particularly crucial for evaluating whether the proposed graph-based approach introduced in Sect.4.3.1 can enhance the performance of similarity measures. For this reason, the distinctiveness of trace classes, according to these structural characteristics, is particularly ensured by the simulated event logs included in the analysis.

SynL1, SynL2, SynL3, and SynL4.

The synthetic event logs *SynL1*, *SynL2*, and *SynL3* are generated using the PLG tool [Bur16]. The traces in these event logs can be clearly distinguished according to the following three criteria: (a) edge types (*loops*, *skips*, or *choices*), (b) length of the edge types (*short* or *long*), and (c) the set of activities ($F_1 = \{A, ..., H\}$ or $F_2 = \{K, ..., R\}$). To test different scenarios for the application of similarity measures, three dif-

ferent classifications of the traces are applied, resulting in three distinct event logs.

In *SynL1*, traces are classified solely according to edge types, resulting in three classes (loops, skips, and choices). This scenario is closely related to the example provided in Fig. 4.1, as the affiliation of traces with a class can sometimes only be detected based on edge types.

In *SynL2*, trace classes are further separated by edge type length, resulting in six classes (loops-short, loops-long, skips-short, skips-long, choices-short, choices-long). This classification allows for the evaluation of similarity measures based on their ability to distinguish between lengths of edge types.

In *SynL3*, trace classes are separated by both edge types and sets of activities, resulting in six classes (loops-F_1, loops-F_2, skips-F_1, skips-F_2, choices-F_1, choices-F_2). This scenario evaluates the effectiveness of similarity measures in differentiating between edge types and distinct sets of activities.

SynL4 extends SynL1 by introducing an additional trace class characterized by combinations of multiple edge types. As a result, SynL4 comprises four distinct classes: loops, skips, choices, and mixed. Traces within each class also vary with respect to the length of the process patterns (i.e., edge types) and the set of activities involved. Compared to SynL1, SynL4 represents a more realistic and structurally diverse scenario, allowing for a more rigorous evaluation of whether similarity measures can accurately identify traces that incorporate heterogeneous structural features.

WABO and PO.

The *WABO* and *PO* event logs were first released as part of the BPI Challenges in 2015 [vD15] and 2019 [vD19]. The datasets originate from real-world business operations.

WABO includes five distinct sublogs representing five different municipalities in the Netherlands. These sublogs document the respective administration processes for handling environmental permits.

PO describes the purchase order handling process within a large multinational corporation located in the Netherlands. The procedure is categorized into four distinct workflow types.

For both event logs, the categorization of traces relies on their structural characteristics, which makes them suitable for the empirical evaluation in the context of process analysis. Given the extensive size of the event logs, random sampling is utilized to decrease the number of traces in proportion to the number of trace classes.

EyeT1 and EyT2.

The event logs *EyeT1* and *EyeT2* are derived from an eye-tracking experiment on process model comprehension [SAAW23, SAAW24] using the EyeMind tool [AALW23]. The eye-tracking data permits analyzing the visual scanpath of a person throughout the experiment. The scanpath consists of a sequence of visual fixations, i.e., moments when a person's gaze rests for a short amount of time on a particular position on a screen. These visual fixations reveal in which order and intensity a person was looking at particular elements of a process model. In *EyeT1* and *EyeT2* each trace reflects the order and number of fixations on process model elements, which were observed during a comprehension task. Thus, each trace might entail structural patterns such as loops, skips, and choices.

Each event log contains 350 traces, which reflect the visual fixations from 44 participants in the experiment. The experiment was designed such that each participant had to answer eight different comprehension tasks, divided into two complexity classes (low and high task complexity). Depending on the task complexity, the participants exhibited different information search and information integration behaviors [SAAW24]. The traces in *EyeT1* are therefore classified according to the eight comprehension tasks, while the traces in *EyeT2* are classified according to the two complexity classes.

A more elaborate discussion on the experiment design and the outcomes of the study is provided in Sect. 6.

4.4.3 Sampling Strategy

Due to the computational complexity of the evaluation measures, particularly the triplet-based evaluation, which has a time complexity of $\mathcal{O}(|L|^3)$, the number of traces per event log is deliberately limited. This approach aligns with the methodology adopted in the comparative study on trace similarity measures by Back and Simonsen [BS23]. While the eye-tracking event logs (EyeT1, EyeT2) are sufficiently small to be used in full, the synthetic logs (SynL1, SynL2, SynL3, SynL4) and the real-world business process logs (WABO, PO) require additional sampling to ensure computational feasibility. To avoid sampling bias, traces are selected randomly. Moreover, sampling is performed in a stratified manner to ensure an equal distribution of traces across classes, thereby preserving the validity and effectiveness of the evaluation (cf. Sect. 4.4.2).

4.4.4 Results

Figure 4.3 presents an overview of the four calculated evaluation measures for the similarity measures discussed in Sect.4.2.3 and the novel similarity measures proposed in Sect.4.3. When examining the Triplet, the most comprehensive evaluation measure included in the analysis (cf. Sect. 4.4.1), it is evident that the novel similarity measures, highlighted in gray, consistently outperform the other similarity measures, with the exception of the event log EyeT1. This trend is partially corroborated by the Nearest Neighbor and Precision@10 measures.

Fig. 4.3 presents a summary of the four computed evaluation measures for the similarity measures discussed in Sect. 4.2.3 as well as the graph-based similarity measures introduced in Sect. 4.3. Looking at *Triplet*, which is the most elaborate evaluation measure included in the evaluation (cf. Sect. 4.4.1), it is evident that the graph-based similarity measures, marked in gray, consistently outperform the other similarity measures, with the exception of event log *EyeT1*. This finding is somewhat supported by *Nearest Neighbour* and *Precision@10*.

It can further be seen that the commonly used *LD* and *N-LD* (cf. Sect. 5.1) are consistently outperformed by the other measures. Additionally, the similarity measures based on 2-gram (*C2* and *J2*) demonstrate consistently strong performance across all evaluation measures and event logs.

The outcome of the evaluation also shows that the aggregation of individual similarity measures often leads to an improvement in the similarity measurement, as indicated by the underlined values in Fig. 4.3. When examining *Triplet*, this is particularly evident for *Agg(C1,C2)* in the context of the two business processes (*WABO* and *PO*).

Furthermore, the similarity measures incorporating edge type features (*Agg(C1,C2,Cg)* and *Agg(J1,J2,Jg)*) perform notably well with respect to

	Nearest Neighbour							
	SynL1	SynL2	SynL3	SynL4	WABO	PO	EyeT1	EyeT2
LD	0.970	0.940	0.967	0.950	0.603	0.873	0.897	0.940
N-LD	0.970	0.933	0.967	0.953	0.600	0.910	0.937	0.971
EMR	0.963	0.917	0.927	0.965	0.200	0.710	0.785	0.871
CMR	0.977	0.957	0.987	0.978	0.203	0.333	0.957	0.971
JMR	0.977	0.957	0.983	0.978	0.194	0.730	0.954	0.971
EF	1.000	**1.000**	0.990	**1.000**	0.457	0.900	0.461	0.662
E1	0.930	0.900	0.913	0.941	0.629	0.860	0.954	0.974
E2	0.997	0.973	0.990	0.981	0.466	0.910	0.885	0.946
E3	0.997	0.970	0.987	0.988	0.366	0.890	0.676	0.811
C1	0.990	0.980	0.987	0.988	0.643	0.890	0.977	0.980
C2	**1.000**	0.990	0.997	0.997	0.611	0.917	0.952	0.966
C3	**1.000**	0.980	0.990	**1.000**	0.609	0.900	0.868	0.917
Cg	**1.000**	**1.000**	0.493	**1.000**	0.206	0.380	0.172	0.653
Agg(C1,C2)	**1.000**	0.993	0.997	**1.000**	0.643	0.917	0.971	0.977
Agg(C1,C2,Cg)	**1.000**	0.997	**1.000**	**1.000**	0.594	0.890	0.957	0.971
J1	0.403	0.290	0.333	0.316	0.646	0.863	0.966	0.980
J2	**1.000**	0.997	**1.000**	0.997	0.611	0.910	0.983	0.991
J3	**1.000**	0.993	**1.000**	**1.000**	0.606	0.920	0.986	0.994
Jg	0.997	0.993	**1.000**	0.994	0.220	0.807	0.960	0.980
Agg(J1,J2)	**1.000**	0.997	**1.000**	0.997	**0.683**	**0.933**	**0.989**	**0.997**
Agg(J1,J2,Jg)	**1.000**	0.997	**1.000**	**1.000**	0.594	0.890	0.957	0.971

	Precision@10							
	SynL1	SynL2	SynL3	SynL4	WABO	PO	EyeT1	EyeT2
LD	0.863	0.758	0.874	0.836	0.398	0.832	0.762	0.851
N-LD	0.875	0.800	0.892	0.861	0.400	0.850	0.865	0.917
EMR	0.871	0.782	0.831	0.853	0.200	0.757	0.621	0.736
CMR	0.941	0.884	0.926	0.932	0.199	0.299	0.903	0.937
JMR	0.946	0.878	0.931	0.930	0.198	0.790	0.901	0.937
EF	0.991	0.974	0.988	0.990	0.359	0.854	0.455	0.706
E1	0.808	0.730	0.830	0.824	0.433	0.857	0.862	0.916
E2	0.917	0.858	0.931	0.927	0.307	0.859	0.708	0.801
E3	0.924	0.874	0.937	0.938	0.251	0.848	0.524	0.673
C1	0.933	0.891	0.919	0.932	0.439	0.875	0.924	0.957
C2	0.962	0.938	0.962	0.967	0.393	0.876	0.842	0.900
C3	0.960	0.936	0.953	0.963	0.386	0.877	0.753	0.835
Cg	0.993	**0.986**	0.482	0.980	0.197	0.600	0.157	0.622
Agg(C1,C2)	0.962	0.938	0.964	0.969	0.424	**0.881**	0.895	0.938
Agg(C1,C2,Cg)	0.995	**0.986**	**0.993**	0.992	0.415	0.871	0.898	0.942
J1	0.367	0.222	0.333	0.379	0.437	0.864	0.905	0.952
J2	0.996	**0.986**	0.990	**0.995**	0.387	0.875	0.958	0.982
J3	0.987	0.968	0.982	0.991	0.379	0.880	**0.966**	**0.987**
Jg	0.964	0.953	0.963	0.943	0.203	0.808	0.924	0.962
Agg(J1,J2)	**0.996**	**0.986**	0.990	0.967	**0.444**	0.880	0.936	0.968
Agg(J1,J2,Jg)	0.995	**0.986**	**0.993**	0.992	0.415	0.871	0.898	0.942

	Triplet							
	SynL1	SynL2	SynL3	SynL4	WABO	PO	EyeT1	EyeT2
LD	0.523	0.545	0.803	0.530	0.504	0.728	0.669	0.584
N-LD	0.418	0.432	0.864	0.431	0.515	0.825	0.873	0.613
EMR	0.537	0.585	0.719	0.564	0.019	0.654	0.662	0.524
CMR	0.428	0.450	0.863	0.439	0.149	0.584	0.891	0.598
JMR	0.428	0.450	0.864	0.439	0.133	0.581	0.885	0.617
EF	0.435	0.469	0.886	0.451	0.525	0.776	0.829	0.610
E1	0.547	0.573	0.806	0.559	0.512	0.747	0.765	0.580
E2	0.563	0.598	0.818	0.586	0.504	0.712	0.680	0.540
E3	0.574	0.616	0.808	0.602	0.496	0.666	0.614	0.513
C1	0.422	0.456	0.870	0.445	0.524	0.853	0.890	0.565
C2	0.438	0.462	0.891	0.458	0.526	0.837	0.822	0.556
C3	0.439	0.461	0.890	0.457	0.407	0.636	0.762	0.533
Cg	**0.816**	**0.917**	0.724	**0.814**	0.152	0.541	0.533	0.577
Agg(C1,C2)	0.434	0.462	0.886	0.456	**0.529**	**0.857**	0.862	0.561
Agg(C1,C2,Cg)	0.653	0.700	**0.910**	0.666	0.524	0.828	0.863	0.582
J1	0.278	0.282	0.631	0.311	0.524	0.799	0.847	0.607
J2	0.419	0.469	0.854	0.437	0.526	0.827	0.915	0.627
J3	0.428	0.468	0.873	0.445	0.407	0.635	**0.919**	0.629
Jg	0.618	0.637	0.782	0.623	0.153	0.716	0.903	**0.663**
Agg(J1,J2)	0.416	0.463	0.849	0.429	**0.529**	0.839	0.878	0.619
Agg(J1,J2,Jg)	0.653	0.700	**0.910**	0.666	0.524	0.828	0.863	0.582

Figure 4.3: The tables show the calculated evaluation measures for the similarity measures on the basis of eight different event logs. The similarity measures proposed in Sect. 4.3 are marked in gray. Values that indicate best-performing similarity evaluations are marked in bold, performance improvement due to aggregations are underlined.

the synthetic event logs (*SynL1*, *SynL2*, *SynL3*, *SynL4*) and *EyeT2*. This

observation is also partially supported by *Nearest Neighbour* and *Precision@10*.

4.5 Discussion

The results confirm the formal comparison of the similarity measures in Sect.4.2.3 and Sect.4.3.2. The empirical evaluation based on a variety of event logs reveals that the combination of individual similarity measures and different sets of structural trace features greatly improves the accuracy of the computed distances. This improvement is in particular evident when considering edge types, which validates the graph-based approach proposed in Sect. 4.3. The inclusion of edge types in the evaluation process provides a more nuanced understanding of trace similarity, highlighting the importance of structural characteristics in similarity measurement.

Moreover, the evaluation underscores the validity of the identified properties P1-P4 and P6-P11 (cf. Table 4.2) as guiding principles for constructing and selecting similarity measures. The structural differentiation of traces based on activities P6, directly follows relations P7, and edge types P8 proves to be especially valuable for both unstructured behavioral processes and structured business processes. These structural properties offer a more detailed and accurate representation of trace similarities, which was not possible based on previous similarity measures proposed in the literature.

The evaluation further reveals that no single similarity measure consistently outperforms the others across all event logs. Instead, different event logs necessitate distinct similarity measures tailored to their specific structural features. One effective approach to address this variability is to aggregate multiple similarity measures and assign different weights to each, emphasizing various features. This can be achieved

by leveraging domain knowledge to determine appropriate weights or by employing automatic weight assignment based on feature learning techniques [BHS13, KB19]. These approaches ensure that the similarity measures are adapted to the unique characteristics of each event log.

Additionally, the evaluation highlights certain limitations of the applied evaluation measures. *Nearest Neighbor* and *Precision@10* exhibit biases when small clusters (comprising two or ten data points) are present, which accounts for the discrepancies observed between these measures and the *Triplet* measure.

A notable limitation of the proposed graph-based approach for comparing process traces is its inability to account for concurrency, i.e., the execution of activities that have no strict causal relationship and may occur in parallel without a defined sequential order. In many real-world processes, such concurrency is common and results in traces where the ordering of concurrent activities may vary across instances, even though the process structure remains equivalent [vdA16]. This variation poses a challenge for similarity measures that assume strict sequentiality, as they may incorrectly interpret such traces as structurally different. However, in the specific context of behavioral analysis based on eye-tracking data, concurrency is not a concern since all recorded events occur sequentially and do not overlap in time. Nevertheless, addressing the issue of concurrency would require additional abstraction in the graph-based analysis introduced in Sect. 4.3.1, potentially through the integration of instance graphs or other concurrency-aware representations, as discussed in Sect. 5.1.

Another potential limitation in the discovery of trace graphs lies in the way edge types are determined by the DFS, specifically, based on the temporal positioning of directly follows relations within a trace. For instance, given the trace $\sigma = \langle A, C, A, B, C \rangle$, according to the DFS, the edge A, C would be classified as *sequence*. However, this

classification may be misleading, as a classification as *skip* would arguably be more appropriate. The classification as *skip* becomes apparent when the same activity pair would occur later in the trace, i.e., $\sigma = \langle A, B, C, A, C \rangle$. This discrepancy can result in misinterpretations of structural similarity across traces, similarly to the issue of concurrency. Nevertheless, it may also accurately reflect temporal variations in a process, e.g., when comparing user behavior processes.

4.6 Summary

This chapter demonstrates the significance of various structural features in measuring the distance between two traces. A graph-based approach is proposed to extract additional high-level structural features from traces, such as loops, skips, and choices. The formal comparison in Sect.4.3.2 and the empirical evaluation in Sect.4.4 indicate that these features can enhance similarity measurement without increasing computational time complexity.

Moreover, the formal properties introduced in Sect. 4.2.1 provide a deeper understanding of which structural trace features are captured by a similarity measure and how the corresponding distance values are computed. These insights can provide better guidance for the selection of a suitable similarity measure for a given process mining task.

Considering edge types as high-level structural features for the calculation of distances between traces opens up numerous opportunities for future research. It allows for a deeper examination of structural similarity, potentially improving various process mining techniques, including trace clustering, conformance checking, event abstraction, event log sampling, change point detection, and variability analysis.

For the analysis of user behavior based on eye-tracking data, edge types can enhance the distinction between traces that demonstrate

varying behavioral patterns, including repetitive fixations, skipping of fixations, or choices between alternative scan paths. This distinction can be particularly valuable for determining if a user is searching for information or integrating information [SAAW23, SAAW24]. Understanding these patterns can help in designing more effective user interfaces or improving the accuracy of behavioral predictions.

Improving the similarity assessment of process traces might necessitate the inclusion of other process perspectives in the analysis, such as those related to resources or data. To achieve this goal, the graph-based measurement approach proposed in this chapter could also be applied to event knowledge graphs, which were initially introduced by Fahland [Fah22]. Thus, the structural analysis would also include additional perspectives of the process beyond the control flow.

5 Structural Process Variety in Business Process Standardization

To provide further empirical validation for the introduced structural process variety measurement framework in Sect. 3.3, this section examines its application within the context of business process standardization.

As outlined in Sect. 3.1, structural variety within business processes is frequently linked to inefficiency and an increased risk of erroneous behavior [Pen03, Men08, MEW10, Tre14, VWM23]. Specifically, a higher number of alternative execution paths within process execution complicates coordination and control. Conversely, structural variety is also recognized as a measure of a process's flexibility and its ability to adapt to changes [Pen03, DRMR13, Tre14, VLS16]. In this context, a greater number of alternative execution paths enables more diverse process behaviors, potentially minimizing the need for adaptations to achieve specific process outcomes. To overcome this apparent trade-off between efficiency and flexibility, *business process standardization*, a common BPM practice, aims to define standardized reference processes [1] that strike a balance between the two [MEW10, Tre14, ABR16, FAM18].

One way to derive a balance between efficiency and flexibility within a reference process is to limit the number of trace variants, while at the same time ensuring sufficient structural variety between the trace

[1] Here, a *reference process* defines constraints for the execution of a process in terms of its control flow. Deviations from the defined control flow are also possible based on adaptations of the reference process. Alternative terms used in the literature are *archetype*, *standard*, *base process*, or *master process* [MEW10, RHB14, Tre14, ABR16].

variants. Such that a process can be efficiently executed (due to the limited number of alternative execution paths), while preserving its adaptability (due to the structural variety among the alternative execution paths). In this context *adaptability*, as defined in Sect. 3.1.1 based on Reichert and Weber [RW12], refers to the ability of a process to react to unforeseen changes and exceptions [2]. The need for adaptability thereby rests on the assumption that not all process behaviors can be anticipated before execution. Therefore, a process should be designed such that structural adaptations during runtime are possible, without incurring high adaptation costs.

The structural properties of the trace variants observed in an event log can serve as a basis for such a standardization approach (a similar approach based on process mining is introduced by Fleig et al. [FAM18]). This can be achieved in the following way. First, to define a standardized process, a process designer must consider a given flat simplified event log L' (cf. Sect. 2.1) as the basis for a reference process. Accordingly, the process designer can restrict the number of trace variants in L' in order to increase the level of standardization, such that $S \subset L'$ is considered as the standardized reference process. Second, to evaluate the adaptability of S, one can calculate the adaptation costs that arise from the removal of all traces $\sigma_i \in L' \setminus S$. For simplicity, this second step assumes that the removed trace variants are eventually executed again and that the defined standardized reference process S needs to be adapted accordingly. However, it would be also possible to consider alternative trace variants $\sigma_i \notin L'$ that have not been observed so far and to check the adaptation costs of S accordingly.

Adaptation costs between sets of trace variants are commonly calculated in process mining based on *optimal trace alignment* [JCBvdA10, vdA16,

[2] This notion aligns with the concept of *flexibility-by-change*, as proposed by Schonenberg et al. [SMR+08b].

CvDW22]. As described in Sect. 2.1.2 optimal trace alignment calculates the minimal costs of editing operations necessary to transform one trace into another. These alignment costs are calculated according to some scoring function, which considers matches, mismatches, and gaps between activities. The scores for the matches, mismatches, and gaps can be adjusted according to the context.

The optimal trace alignment is derived in a similar way as the edit distance, which is calculated based on the minimal number of insertions, deletions, and substitutions of activities needed to transform one trace into another (cf. Sect. 2.1.2). It is indeed the case that edit distance and optimal trace alignment yield equal outcomes in some cases (a proof is provided by Sellers [Sel74]). This is in particular important when analyzing the relationship between structural process variety, which is derived based on edit distance, and adaptability, which is derived based on alignment costs.

On the basis of the structural process variety and the adaptation costs, one can make the following observations. As shown in Fig. 5.1, an increase in the *reduction of trace variants* within a reference process (depicted on the x-axis) leads to a decrease of its *structural process variety* (depicted on the y-axis). This is illustrated by the decreasing function representing *standardization*. At the same time, the *reduction of trace variants* also requires higher adaptation costs if the *structural process variety* of the executed trace variants increases. This is illustrated by the increasing function representing *adaptation costs*.

According to this observation, the analysis of structural variety can support the design of standardized reference processes in three ways. First, the evaluation of process variety based on some event log L' can serve as a proxy for adaptability. Second, it shows how individual trace variants in L' contribute to the overall structural process variety and how they affect adaptation costs. Third, it reframes business

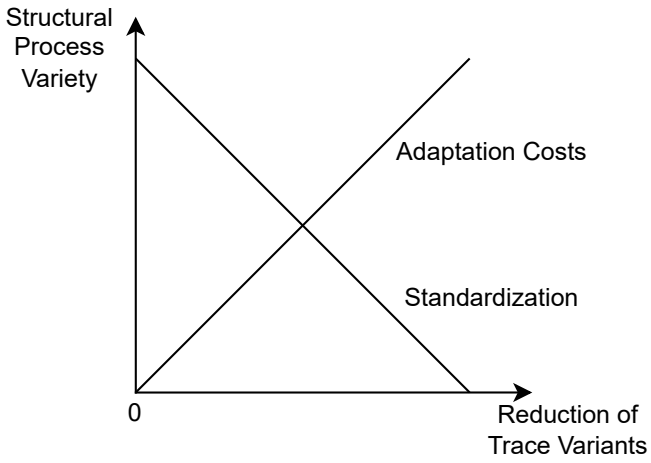


Figure 5.1: Trade-off in business process standardization between flexibility (measured in terms of structural process variety on the y-axis) and efficiency (measured in terms of the reduction of trace variants on the x-axis). The diagram also shows the contrary relation between standardization and adaptation costs.

process standardization as a multi-objective optimization problem, necessitating the identification of an optimal balance between efficiency and flexibility.

The empirical evaluation in this chapter therefore seeks to answer the following main research question:

RQ: Can the proposed measurement framework for structural process variety provide guidance for process standardization by deriving more efficient and flexible reference processes than alternative standardization approaches?

To answer this question, the subsequent section first identifies some alternative approaches and criteria for the design of standardized reference processes, which have been proposed in the literature. After-

wards, the proposed measurement framework in Sect. 3.3 is evaluated according to its ability to derive standardized reference processes, which strike a balance between efficiency and flexibility.

5.1 Related Work

The general idea of *business process standardization* is to align similar business processes in a uniform way [MEW10, RDGvW15]. The creation of uniform processes aims to reduce variety in process execution in order to enhance efficiency, improve quality, and facilitate communication [MEW10, Tre14, RDGvW15]. Several empirical studies could indeed show the positive effects of business process standardization on process performance aspects such as time, costs, quality, and control [FKLM99, JVD00, BGJH09, WKBW09, MEW10].

Nevertheless, it has been recognized that business process standardization also requires maintaining some process variety in order to ensure flexibility and autonomy among different business units [FKLM99, Tre14, RDGvW15, WSMF18]. In an empirical study, Frei et al. [FKLM99] could show that the process variety is particularly necessary to serve different customer needs and thus should be the focus for process improvement, rather than focusing on a single standardized process variant [3].

Tregear [Tre14] points out that the trade-off between efficiency and flexibility especially occurs when a global organization aims to define a standardized process, which needs to be executed across multiple organizational units located in different countries. Tregear [Tre14] highlights several reasons why local adaptations of a global reference process may be required, including:

[3] In the study, which was conducted in the finance industry, a process variant refers to a specific financial service provided by a bank to a customer, such as *opening a checking account*, or *opening a small business loan account*.

- legislative requirements, which might stem from financial regulations, taxation regimes, or other regulations
- local market conditions defined by regional culture, customer expectations, market maturity, or competitive landscape
- individual preferences of business process stakeholders
- resource constraints according to local conditions
- legacy systems, that cannot be merged with other IT systems
- concept drift as a result of small process changes over time (as described in Sect. 3.1.1)
- knowledge intensive process with no frequently reoccurring structure.

As discussed in Sect. 3.1.1, these different flexibility needs can be addressed at a structural process level [SMR+08b, RW12].

There exist indeed several studies, which have identified structural process properties as a relevant success factor for business process standardization [RSM+10, Tre14, RDGvW15, WSMF18, FAM18]. Based on three case studies, Rosenkranz et al. [RSM+10] point out that the structuredness of a process is a key success factor for business process standardization. Furthermore, several indicators for the evaluation of business process standardization based on the control flow have been identified [RDG+15, WSMF18], such as the occurrence of process exceptions, the reoccurrence of activities, as well as sequences of activities.

Similarly, Vidgof et al. [VWM23] could show that different complexity measures, that are closely related to structural process variety (cf. Sect. 3.1.2), can have a significant impact on process performance. Hence, the authors suggest that these measures can also provide guidance for process standardization initiatives.

Furthermore, Fleig et al. [FAM18] propose a method, which supports business process standardization by matching processes with predefined best-practice processes, according to their structure. This method, however, does not involve any structural variety considerations.

Overall, it appears that there do not exist structural process measures, which have been explicitly applied for the analysis of process standardization. The identified complexity measures by Vidgof et al. [VWM23], which partially consider structural process variety, have so far only been considered in relation to efficiency, but not to flexibility. Furthermore, there appear to exist no studies considering the ranking of trace variants according to their marginal contribution to the overall flexibility of a process.

The subsequent section evaluates to which extent the identified shortcomings can be addressed by using the measurement framework for structural process variety introduced in Sect. 3.3.

5.2 Evaluation

The evaluation is based on four real-life event logs derived from four different business organizations. Two different approaches for process standardization are compared: (a) based on the proposed measurement framework for structural process variety, and (b) based on the relative frequencies of the process variants in the event log, which is a common practice for business process standardization [RHB14, Tre14, FAM18].

5.2.1 Evaluation Method

The comparison of the two standardization approaches involves the ranking of the trace variants within the respective event logs according to (a) their marginal contribution to structural process variety, and

(b) their relative frequencies. Subsequently, the rankings are compared based on a step-wise reduction of trace variants, thus generating multiple standardized reference processes. The step-wise reduction of trace variance implies at each step an incremental increase in efficiency. However, to also evaluate the impact of the step-wise reduction on the process's flexibility, the increase in alignment costs is additionally considered at each step. The alignment costs are determined by calculating the optimal alignments between the removed trace variants and the remaining trace variants in the reference process. The optimal alignment is calculated based on a cost of 1 for gaps and mismatches, and rewards of 0 for matches (cf. Sect. 2.1.2).

The step-wise reduction respectively starts with (a) the trace variant that contributes the least to the structural process variety, and (b) the trace variety with the lowest frequency. When trace variants have equal variety or frequency values, one of them is chosen at random.

To quantitatively compare the two selection methods, two distinct evaluation measures are employed: the approximate integral of the alignment cost curve using the trapezoid rule [Gau11], i.e., the *area under the curve* (AUC), and the *point of intersection* (PoI) between the alignment cost curve and the variety curve. The AUC serves as the primary evaluation measure, as it quantifies the increase of alignment costs in relation to the reduction of trace variants. A lower AUC value signifies a slower increase in alignment costs, which is an indicator of improved process standardization.

Additionally, the PoI can be considered as an equilibrium point between standardization and adaptation costs. PoI is measured in terms of the number of reduced trace variants. Hence, a higher PoI indicates an improvement in terms of efficiency and flexibility.

	Activities	Trace Variants	Origin
WABO 2014	465	4.245	Administration
BPI 2015	29	4.209	Administration
BPI 2017	26	15.930	Administration
PO 2019	36	10.207	Administration

Table 5.1: Overview of the event logs including the number of activities, the number of trace variants, and their origin.

5.2.2 Datasets

The four real-life event logs that are considered for the evaluation are depicted in Table 5.1. The event logs are selected due to their differences in complexity, in terms of the number of activities and the number of trace variants. Furthermore, the event logs represent administrative business processes, where process standardization could potentially be applied.

The four event logs considered (WABO 2014, BPI 2015, BPI 2017, PO 2019) correspond to the following processes:

- WABO 2014 [Bui14b] documents the execution phase of the building permit application process across various municipalities in the Netherlands. The process is consistent among these municipalities, meaning that the different event logs correspond to the same activities (cf. Sect. 4.4.1).
- BPI 2015 [vD15] is derived from WABO 2014 and was released for the BPI challenge 2015.
- BPI 2017 [vD17] relates to a loan application process of a Dutch financial institution, with data comprising applications submitted through an online system.

- PO 2019 [vD19] details the purchase order handling process of a multinational coatings and paints company, encompassing four distinct workflows for purchase order items. Each workflow varies in terms of invoices, goods receipts, and purchase values.

5.2.3 Results

The structural variety and alignment cost curves derived (a) from the framework and (b) the frequency-based selection are illustrated in Fig. 5.2. The variety curves generated by the framework exhibit greater concavity across all four logs, indicating a slower decrease in structural variety relative to the reduction of trace variants.
The evaluation measures are presented in Table 5.2. The framework-based reduction of trace variants consistently yields superior results for both the AUC and the PoI across all four logs, in comparison to the frequency-based reduction.

Overall, the selection of trace variants based on structural process variety outperforms the selection based on relative frequencies, yielding standardized reference processes that excel in efficiency and flexibility.

5.3 Discussion

The evaluation in the previous section has shown that structural process variety, calculated based on the proposed framework in Sect. 3.3 can provide valuable guidance for process standardization efforts and surpasses the selection based on relative frequencies. The selection of trace variants according to their marginal contribution to the overall structural variety therefore provides a promising new direction in BPM research, as well as business process standardization practice.

Nevertheless, there are some limitations of the introduced approach to business process standardization. First of all, it is questionable whether

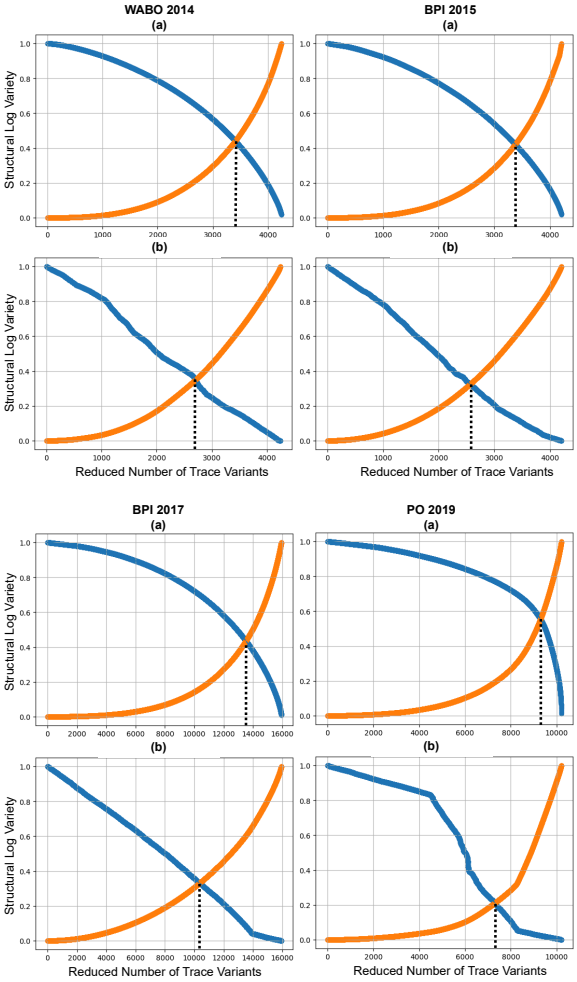


Figure 5.2: The structural variety curve (blue) and the alignment costs curve (orange) in relation to the reduced number of trace variants (x-axis) and the structural log variety (y-axis). The diagrams show the results for the evaluation based on four different real-life logs, when ranking the trace variants based on (a) structural variety and (b) relative frequency.

Log file	Selection Approach	AUC	POI
WABO	Structural Variety	941,44	3406
	Frequency	1296,23	2698
BPI 2015	Structural Variety	891,98	3377
	Frequency	1278,63	2565
BPI 2017	Structural Variety	2879,23	13516
	Frequency	4416,32	10356
PO 2019	Structural Variety	1698,93	9295
	Frequency	1858,98	7300

Table 5.2: Comparison of the two trace variant selection approaches based on (a) structural variety and (b) frequency.

efficiency can be approximated by the number of trace variants and flexibility by structural process variety. However, several studies have shown that the process structure, in terms of the control flow, is indeed an important aspect of business process standardization. Hence, the approach proposed in this section can serve as a starting point to think about additional structural process properties that are relevant for the evaluation of flexibility and efficiency, such as specific control-flow patterns [RW12]. These additional process properties could then be integrated into the measurement of structural process variety and optimal trace alignments.

Another limitation is that so far only structural properties of the traces have been considered, neglecting other process aspects such as resources or data. Future work could integrate these additional perspectives into the framework by using different notions of trace variant and distance. An alternative way to integrate additional process aspects is proposed by Kabierski et al. [KNGW21].

Additionally, some uncertainty persists regarding the derived variety value due to the application of the distance measure and the cluster-

ing algorithm. The clustering procedure merely provides an approximation of the variety. However, several evaluation measures can be applied to assess the quality of a clustering outcome and therefore enhance the confidence in the variety evaluation. Given a ground truth, i.e., the actual clusters are known, Han [HKP12] lists four commonly applied evaluation measures: cluster homogeneity, cluster completeness, rag bag, and small cluster preservation. Additionally, silhouette coefficient is a common evaluation measure, if the ground truth is not known [HKP12].

Similarly, there exists uncertainty concerning the ranking of trace variants. Due to the ultrametric property, the two greatest distances between three trace variants are always equal (cf. Sect.2.3.2). Consequently, even for a small set of three trace variants, multiple ranking possibilities exist. While the framework can guarantee a unique structural variety curve (cf. Sect.3.3), for the ranking this is not the case. In practice, involving experts in the ranking decision, who possess additional insights on the trace variants beyond their distance values, could address this issue.

As the evaluation could show, the selection of trace variants based on structural variety surpasses the selection based on relative frequencies. One reason for this is that the relative frequency of trace variants is also not considered by the introduced evaluation measures AUC and POI. This can be reasonable for business processes where flexibility is more important than frequency, such as in service work [Pen03]. Additionally, the real-life logs used in the evaluation mainly contain infrequent behavior, with the share of trace variants occurring only once being greater than 80% in all event logs (WABO - 99%, BPI 2015 - 92%, BPI 2017 - 87%, PO 2019 - 81%). Nevertheless, frequency can be important in some application scenarios, such as process data sampling [KNGW21]. If specific structural characteristics occur more frequently,

this should be reflected in the sampling of traces. One way to integrate frequencies into the framework is to adjust the calculation of pairwise distances, such that the distance increases with the frequency of required edits. An alternative approach would be to consider the different variety dimensions separately from each other based on multiple measures to avoid contradicting results (cf. Sect.2.3.2).

Regarding the application of the introduced process standardization approach based on structural process variety, it should be noted that the approach so far does not consider interdependence between trace variants. In reality, this is not necessarily the case. Therefore, trace variants cannot simply be removed based on their marginal contribution to structural variety. Future research could investigate how a variety measure could handle interdependence among traces. Additionally, a qualitative evaluation of the framework would be necessary to further validate its applicability for process standardization in practice.

Finally, the proposed process standardization approach has so far only been considered from a backward-looking perspective, based on already observed process variants. However, the approach could also be applied in forward-looking scenarios, such as simulations [vdA18, KSR23]. This would allow a process designer to evaluate the adaptation costs of a standardized reference process based on alternative trace variants not already contained in a log.

5.4 Summary

This chapter shows that structural process variety can indeed provide guidance for business process standardization, by deriving standardized reference processes that provide more efficiency and flexibility than alternative standardization approaches. One reason for this is that flexibility has generally been neglected when trying to measure

the level of process standardization. Hence, this chapter introduced a new way of deriving standardized reference processes, as well as novel evaluation measures for standardization efforts based on AUC and POI. Future research can build on this novel approach, e.g., by introducing additional process perspectives or by considering the interdependence among the trace variants.

6 Structural Process Variety in User Behavior Analysis

To seek further validation for the concept and measurement of structural process variety, this chapter revolves around the analysis of user behavior through eye-tracking. Similarly to business processes, user behavior can be represented as traces, which reflect sequences of activities that are observed from the interaction of users with an information system. A common example for such interaction is click-streams [WZT+16, FVLK+17], which document the sequence of clicks a user performs when interacting with a visual user interface. Another example is visual scan-paths derived from eye-tracking [HNA+11], which document a series of a user's visual fixations while looking at a user interface.

As discussed in Sect. 2.3, the concept of variety can differ depending on the context in which it is applied. In Chapter 5, it was shown that structural variety within business processes is often linked to improvements in efficiency and adaptability. In contrast, when analyzing user behavior, the structural variability of visual scan-paths reflects underlying cognitive processes. These patterns can provide insights into specific phenomena such as cognitive impairments [SCBS08], aesthetic evaluations [KSDK14], and more broadly, the assimilation of information [HNA+11, PM13, SAAW24].

Although structural process variety has been acknowledged as a valuable source of insight into human behavior, existing studies predominantly rely on entropy, i.e., a frequency-based measure of variability

(cf. Sect. 2.3). However, as discussed in Sect. 3.3.1, entropy fails to capture structural distinctions between behavioral traces, such as loops, skips, and choices. These structural patterns may reflect meaningful user behaviors, including information skimming (skips), sustained attention to specific visual regions (loops), or the differentiation between relevant and irrelevant content (choices) [PM13].

To demonstrate that a more nuanced consideration of structural properties can enhance the analysis of user behavior through eye-tracking data, this chapter presents a comprehensive eye-tracking study designed to collect empirical data on user interactions during comprehension tasks. Building on this empirical foundation, the chapter further conducts a concept drift analysis [BvdAŽP11], employing the structural similarity measures introduced in Chapter 4 to identify change points in user behavior. These change points indicate transitions from information search to information inference, thereby providing deeper insights into behavioral processes and enabling more effective user support mechanisms.

Investigating Process Model Comprehension Tasks

Solving comprehension tasks that are linked to process models is a fundamental requirement for both technical and managerial activities [Fig17, Zug13]. In technical contexts, tasks such as model maintenance, refinement, and re-engineering depend on a clear understanding of the model's structure and semantics, as well as the ability to apply suitable methods and tools for adaptation or extension [Zug13]. From a managerial perspective, process model comprehension facilitates requirement elicitation and enhances communication between domain experts and IT professionals [Fig17, IGRR09]. To effectively respond to evolving business needs, process modelers must first grasp the current configuration of a process model, which is essential for driving

improvements in performance and productivity. Although automated techniques such as process mining [vdA16] and generative machine learning approaches [vDMKR23] can produce process models, human understanding remains indispensable for interpreting, validating, and refining these models.

While process model comprehension has been extensively studied within the field of BPM, existing research has primarily concentrated on the graphical representation of process models. This focus has often overlooked the task-oriented perspective, which considers the specific types of tasks that process models are intended to support [MM21]. The eye-tracking study presented in this chapter addresses this gap by investigating two prevalent task types: ***local tasks***, which involve analyzing a specific segment of the model, and ***global tasks***, which require synthesizing information across multiple segments. This distinction is relevant across various domains. In software engineering, for example, local tasks pertain to understanding individual microservices, whereas global tasks involve analyzing the interdependencies among multiple microservices [MFF+23]. Similarly, in logistics, tasks may focus on isolated sub-processes or span several interconnected sub-processes [BMPR12].

Modularization, the decomposition of complex systems into smaller, interconnected modules, is a common strategy for enhancing the comprehensibility of information systems. However, empirical studies indicate that modularization does not uniformly lead to improved performance in comprehension tasks, due to the competing cognitive effects of ***abstraction*** and ***fragmentation*** [RM08, Zug13, WPP+22]. Abstraction facilitates comprehension by enabling information hiding and promoting pattern recognition, thereby reducing cognitive load. In contrast, fragmentation imposes additional cognitive demands, as users must integrate information distributed across multiple modules.

This trade-off affects task performance differently depending on task type: local tasks tend to benefit from abstraction, making them easier to complete, whereas global tasks are hindered by fragmentation, increasing their complexity. The interplay between modularization, task type and cognitive processing in model comprehension remains insufficiently understood and requires further empirical investigation [Zug13, RM08, FL15, PMR16].

Additionally, further investigation is required to better understand the cognitive processes involved in comprehending process models, particularly to understand how information is processed across different task types [MM21, BSP19, RRvdW14]. Although prior research has demonstrated that task type significantly influences information search and processing behavior [Vak99, HAC13, CWL+19], this relationship has not been thoroughly examined in the context of process model comprehension [Fig17, MM21, BSP19]. Analyzing the cognitive steps involved in local and global tasks can shed light on the effects of abstraction and fragmentation, thereby enabling more effective, context-sensitive support during task execution. Two key cognitive operations in comprehension tasks are ***search*** and ***inference*** [MM21, BSP19, LS87]. Search involves identifying and isolating relevant information from irrelevant content [Wol94], while inference refers to the constructive cognitive process through which new understanding is derived from the gathered information [LS87]. A deeper understanding of these processes can, for example, inform the development of dynamic support mechanisms, such as context-aware information highlighting [PMR16, HAC13], and intelligent search assistance [Vak99, CWL+19].

Subsequently, the guiding research questions for the study of process model comprehension can be formulated as follows:

RQ1: *How do users' comprehension and cognitive load differ between local*

and global tasks?

RQ2: *How do users' search and inference behavior differ between local and global tasks?*

The eye-tracking experiment designed to address these research questions employs a modularized process model, developed using the fragment-based modeling approach [HW16], alongside a series of local and global comprehension tasks. The experiment evaluates the influence of task locality on users' comprehension performance, cognitive load, visual search behavior, and inference strategies. Both conventional model comprehension measures and subjective cognitive load assessments are utilized, following established methodologies in the literature [Fig17, Zug13]. In addition, objective cognitive load indicators derived from eye-tracking data are incorporated to provide a more nuanced understanding. This multifaceted measurement framework supports a comprehensive empirical analysis, thereby increasing the reliability of the findings. The results highlight the critical role of task type for process model comprehension. Global tasks are associated with reduced model comprehension and elevated cognitive load compared to local tasks, an effect attributed to model fragmentation. Furthermore, global tasks involve longer and more visually complex search and inference phases. Notably, the fragmentation effect is more pronounced during inference than during search.

Considering Structural Process Variety to Detect Concept Drifts in User Behavior

Building on the gained insights from part one, concept drift analysis can help to further deepen the analysis of user behavior, specifically with regard to search and inference behavior. Concept drift in the context of process mining, generally refers to the detection and characterization of second-order process changes [BvdAŽP11], which can be

especially relevant in dynamic, user-driven processes [ŽPG16]. Detecting and analyzing the transition from search to inference can thereby enable more accurate modeling of user behavior, support the development of adaptive systems that respond to evolving user needs, and enhance the personalization of interfaces and assistance tools [ŽPG16].

Applying concept drift in process mining necessitates the transformation of process data into a time series format, which in turn requires the use of appropriate quantitative measures to capture temporal dynamics [BvdAŽP11]. In this context, the present chapter addresses the following research question:

RQ3: *Does the consideration of structural process variety at trace level improve the detection of concept drift in user behavior?*

The conducted concept drift analysis demonstrates that the structural similarity measures introduced in Chapter 4 effectively enhance the detection of concept drift in user behavior. This finding is consistent across both global and local tasks. However, the performance of these similarity measures is influenced by the temporal resolution of the analysis, particularly the length of the time window defined by the number of visual fixations. Longer time frames tend to capture a greater number of structural patterns, which can improve the sensitivity of change point detection but may also introduce additional complexity.

This chapter is structured into the following sections. Sect. 6.1 provides some background on process model comprehension, eye-tracking analysis, and concept drift analysis. Sect. 6.2 introduces related work on process model comprehension, cognitive information processing, and the detection of concept drift in user behavior. Sect. 6.3 outlines the applied research method. The outcomes of the analysis are presented in Sect. 6.4. Subsequently, Sect. 6.5 offers a discussion on these findings, which is followed by a summary provided in Sect. 6.6.

6.1 Background on Process Model Comprehension

The analysis presented in this chapter builds on theory related to abstraction and fragmentation (Sect. 6.1.1), process model modularization (Sect. 6.1.2), comprehension and cognitive load (Sect. 6.1.3), information processing (Sect. 6.1.4), and concept drift analysis (Sect. 6.1.5).

6.1.1 The Effects of Abstraction and Fragmentation in Modularized Process Models

The impact of modularization on business process model comprehension remains inconclusive, with studies reporting both positive and negative effects [Fig17, Zug13]. This inconsistency is often attributed to the dual cognitive effects introduced by modularization, i.e., abstraction and fragmentation, which arise from the spatial separation of model components [Zug13, RM08, TRV+16].

Abstraction results from decomposing a process model into smaller, manageable sub-components, allowing users to concentrate on task-relevant information while filtering out irrelevant details. This can enhance comprehension by reducing cognitive load and supporting pattern recognition, particularly in complex models [PMR17, MRvdA10]. Consequently, modeling guidelines advocate for the decomposition of large models to improve clarity and facilitate understanding [MRvdA10].

Conversely, ***fragmentation*** can undermine these benefits when essential information is dispersed across multiple modules. Navigating between spatially separated fragments may induce the split attention effect [SC94, Zug13], where users must mentally integrate information from different locations. This increases cognitive effort, risks memory decay, and complicates the comprehension process. Such effects are likely to manifest in users' visual search and inference behavior.

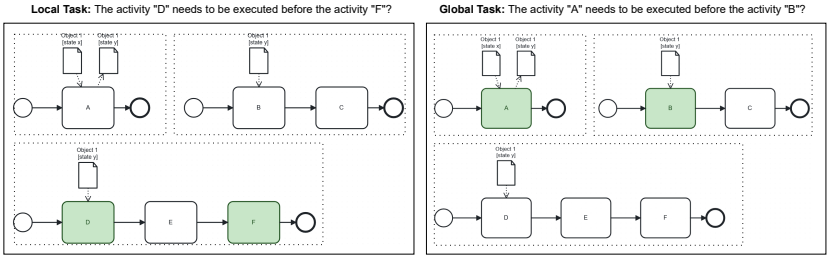


Figure 6.1: Comparison between a local and global task based on a modularized process model consisting of three fragments. While both task types benefit from abstraction, the global task is additionally impeded by fragmentation, since the two relevant activities “A" and “B" are distributed among two fragments.

The tension between abstraction and fragmentation becomes particularly evident when comparing local and global tasks. Local tasks involve a single module, allowing users to benefit from abstraction without the drawbacks of fragmentation. In contrast, global tasks require integration across multiple modules (cf. Fig. 6.1), making them more susceptible to fragmentation-related challenges. This aligns with cognitive fit theory [Ves91, DG01], which posits that the alignment between task requirements and problem representation influences problem-solving performance. The fragmented structure of global tasks weakens this alignment, potentially impairing model comprehension, increasing cognitive load, and altering visual processing behavior.

Building on prior research on modularization and process model comprehension [Zug13, RM08, FL15, PMR16], the eye-tracking study presented in this chapter offers a more nuanced understanding of the cognitive effects of abstraction and fragmentation. This is achieved by: (1) focusing on horizontal rather than vertical modularization, (2) employing a diverse set of measures to assess both comprehension performance and cognitive load, and (3) analyzing user information pro-

cessing through the lens of search and inference behaviors across local and global task types. The following sections introduce the theoretical constructs and empirical measures used to evaluate the influence of task locality on process model comprehension within modularized process models.

6.1.2 Modularization in Process Modeling

Modularization refers to the systematic decomposition of a system into interconnected components, or modules, each characterized by a degree of internal cohesion and functional independence [Par72]. Within the domain of process modeling, three primary forms of modularization have been identified: vertical, horizontal, and orthogonal [WPP+22, LRWM+11].

- ***Vertical modularization*** organizes a process into a hierarchical structure of sub-processes, enabling hierarchical abstraction and control-flow management [WPP+22, LRWM+11].
- ***Horizontal modularization*** partitions a process into functionally related fragments that may be executed either sequentially, concurrently, or hierarchically, without enforcing a strict hierarchical structure [WPP+22, DG01].
- ***Orthogonal modularization*** addresses cross-cutting concerns, such as security or compliance, that span multiple parts of a process. These concerns are encapsulated in separate modules to maintain clarity and modular integrity [WPP+22, DG01]. For instance, identity verification steps repeated across a process may be modularized independently for security purposes.

Among these, horizontal modularization is particularly well-suited for investigating the cognitive implications of task locality, as it allows for the separation of process components without imposing hierarchical

constraints or focusing on cross-cutting concerns. Prior research suggests that horizontal modularization can enhance model comprehension by reducing complexity and cognitive load [WPP+22]. However, this study specifically examines how horizontal modularization influences comprehension in tasks that are either local (focused on a single module) or global (requiring integration across multiple modules). To operationalize this form of modularization, the eye-tracking study adopts the fragment-based modeling approach proposed by Hewelt and Weske [HW16].

The fragment-based modeling approach is based on the widely adopted Business Process Model and Notation (BPMN) standard [HW16]. The modeling approach enables the integration of the case-based modeling paradigm, which supports flexible and adaptive process execution [BMPR12, vdAWG05]. The modeled process fragments (sub-processes) are interconnected through data constraints, which define the input and output conditions for individual process activities. As a result, the execution order of fragments is not governed by hierarchical structures. At runtime, fragments can be executed independently, provided that the specified data constraints are satisfied. These constraints are based on data objects, each characterized by a data type and a state. Throughout process execution, data objects may undergo state transitions, which are modeled alongside the process fragments using a labeled transition system, referred to as a lifecycle.

Fig. 6.2 illustrates an example of the fragment-based modeling approach as applied in the eye-tracking experiment. In this example, the fragments “F1_Offload_Container” and “F1_Scan_Container” are linked via the data object “Container”. Once the activity “Load container onto the forklift truck” in the first fragment is executed the state of the container changes to “lifted’, enabling the execution of the second fragment. Notably, the second fragment depends on two

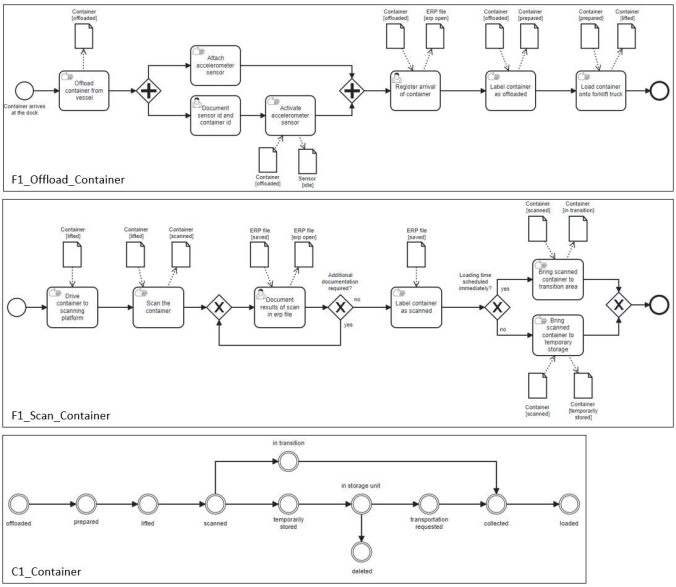


Figure 6.2: Example of a horizontally modularized process model following the fragment-based approach.

data objects: "Container" and "ERP file". The associated lifecycle "C1_Container" further shows that the container reaches its final state when it is marked as "loaded".

Beyond the fragment-based approach, several other modeling techniques align with the case-based modeling paradigm. These include methods that connect process fragments through event-based mechanisms rather than data objects (for an overview see Krumeich et al. [KWWL14]) as well as adaptations of alternative modeling languages such as (colored) Petri nets [FR21]. The decision to adopt the fragment-based approach proposed by Hewelt and Weske [HW16] for the eye-tracking experiment is motivated by the popularity of the BPMN standard. Furthermore, BPMN offers a well-defined execution semantics

and an intuitive visual syntax, making it accessible even to users with limited modeling experience. Additionally, the integration of data objects to interconnect fragments significantly enhances BPMN's expressiveness, enabling more accurate representations of real-world process behavior.

6.1.3 Comprehension and Cognitive Load

This section introduces the theoretical constructs and empirical measures used to examine the effects of abstraction and fragmentation on *model comprehension* and *cognitive load*.

Comprehension. Comprehension refers to the cognitive process through which individuals interpret an external representation, such as a process model, and construct a corresponding mental model [KSP05]. A wide range of comprehension measures has been developed to assess how various representational factors (e.g., presentation format, model complexity, activity labeling, and task type) influence a user's ability to understand the information conveyed by a process model [Fig17]. For the eye-tracking study, two widely used measures are employed: *comprehension accuracy*, which reflects the correctness of task responses, and *comprehension efficiency*, which considers the time required to complete the task.

Cognitive load. According to cognitive load theory, task difficulty increases as individuals approach the limits of their working memory capacity, which can impair performance and elevate the likelihood of errors or suboptimal decisions [CZW+16, PTTVG03]. In the context of process model comprehension, elevated cognitive load may hinder understanding and increase the risk of misreading process behavior. Cognitive load can be assessed through both subjective and objective measures [CZW+16, HNA+11, WFR21]. A common subjective method

involves self-reported *perceived difficulty*, captured by Likert-scale ratings [SAS11].

To complement subjective assessments, the eye-tracking study also incorporates objective measures. Eye-tracking enables the identification of visual and behavioral patterns that may not be evident through verbal reports alone [HNA+11]. A key measure is *average fixations duration*, which represents the mean duration of fixations, i.e., periods of relatively stable gaze, within a given time window. Shorter fixations are typically associated with scanning behavior, while longer fixations indicate deeper cognitive processing [HNA+11].

Additionally, a *physiological measure* of cognitive load is employed. Physiological responses to cognitive demands are regulated by the autonomic nervous system, with the sympathetic division activating the body under high mental load and the parasympathetic division promoting relaxation during low demand [WFR21, RL15]. One such response is pupil dilation, which can be quantified using the *low/high index of pupil activity (LHIPA)* [DKG+20]. This index differentiates between low- and high-frequency pupil oscillations and is negatively correlated with cognitive load, i.e., lower LHIPA values indicate higher cognitive demands.

6.1.4 Search and Inference Phases

In addition to the consideration of local and global tasks, the eye-tracking study also differentiates between the search and inference phases of task execution to capture the dynamics of information processing during process model comprehension.

Search. The search phase involves the identification and discrimination of relevant from irrelevant information [Wol94]. In the context of process model comprehension, this corresponds to distinguishing

between *task-relevant* and *task-irrelevant* activities within the model [PM13]. The search process is influenced by both the task type and the search strategy employed [BP04, BPW07, GH11]. Given that search consumes cognitive resources such as time and cognitive effort, individuals typically terminate the search once they perceive that sufficient information has been gathered [BP04]. Here, sufficiency refers to the perceived completeness and correctness of the information required to perform the task, which presupposes that relevant elements can be clearly identified. To determine when to stop searching, individuals apply *cognitive stopping rules* [BP04]. For well-structured and decomposable tasks, a common strategy is the use of *mental lists*, i.e., internal checklists of items that must be located before concluding the search [BPW07]. In process model comprehension, this strategy may be employed when users are asked to explain the relationships among a set of activities, prompting them to search for and mentally verify the presence of each relevant activity before transitioning to the next phase.

Inference. The inference phase involves the cognitive integration of identified information with prior knowledge to derive new insights [LS87]. This process is closely associated with reasoning [KHH00]. In the context of diagrammatic representations such as process models, inference encompasses both *perceptual* processes (e.g., recognizing relevant elements) and *conceptual* processes (e.g., drawing conclusions based on relationships among elements) [KHH00]. According to multimedia learning theory [May89], inference follows the selection of relevant information and involves organizing and integrating this information into a coherent mental model.

To investigate the characteristics of search and inference phases, the study utilizes eye-tracking data to capture visual behavior (see Fig. 6.3). Eye-tracking provides valuable insights into cognitive processing that may not be accessible through verbal reports alone [HNA+11]. A key

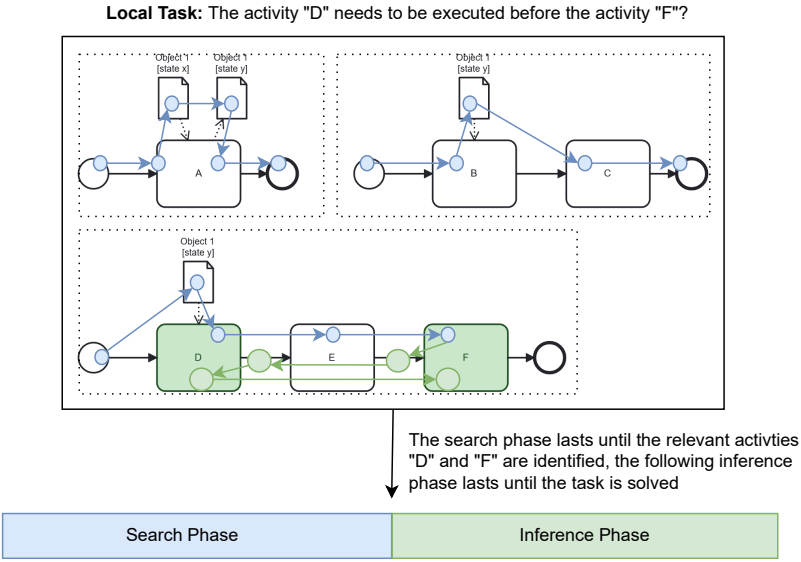


Figure 6.3: Distinction between search and inference phase. The blue dots and arrows depict a scan-path during search and the green dots and arrows depict a scan-path during inference.

measure is fixation duration. *Short fixations* (typically < 250ms) are indicative of superficial processing and are commonly associated with information search, whereas *long fixations* (≥ 500ms) suggest deeper cognitive engagement, characteristic of inference processes [GH08]. In addition to fixations, *saccades*, i.e., rapid eye movements between fixations, offer further insight into cognitive states. The *average saccade amplitude*, which measures the average distance between fixations, tends to decrease during intensive search, reflecting focused attention and cognitive effort [HNA+11].

An additional key concept for analyzing information processing is the scan-path. A *scan-path* represents the sequential pattern of fixations made by a user while visually engaging with a stimulus, such as a

process model [HNA+11]. In eye-tracking research, scan-path analysis often focuses on whether the visual trajectory includes multiple areas of interest (AOIs), i.e., predefined regions within a stimulus that are relevant to the research objective [HNA+11]. In the context of process model comprehension, AOIs typically correspond to individual model elements, such as activities or gateways. A scan-path includes both first-time visits and revisits to these AOIs.

One important measure derived from scan-paths is *scan-path precision*, which quantifies the proportion of fixations on task-relevant AOIs relative to the total number of fixations on the entire artifact [PM13]. This measure provides insight into the degree of attentional focus on relevant information. During the search phase, users are actively identifying relevant elements among distractors, often resulting in a lower scan-path precision due to dispersed fixations. In contrast, during the inference phase, after relevant elements have been identified, users tend to concentrate their attention on these elements, leading to higher scan-path precision.

Another related measure is the *entropy* of fixations, which measures the uncertainty or variability in fixation distribution across AOIs [HNA+11]. Low entropy values indicate concentrated attention on a limited set of AOIs, typically associated with inference, while high entropy values reflect exploratory behavior characteristic of the search phase.

6.1.5 Concept Drift Analysis

Building on the notion of a scan-path, concept drift analysis can be employed to detect changes in the sequence of visual fixations, which may indicate shifts in cognitive processing patterns. For instance, during information search, scan-paths typically exhibit linear reading patterns, whereas during information inference, they tend to become more complex, often involving repeated transitions between Areas of Inter-

est (AOIs) (cf. Sect. 6.1.4). These structural changes in scan-paths can thus serve as indicators of behavioral transitions, such as the shift from information search to inference.

The term *concept drift* originates from the fields of machine learning and data mining, where it refers to changes in the relationship between input features and target variables over time [GŽB+14, LLD+18]. This phenomenon is particularly relevant in online supervised learning scenarios, where models trained on historical data may lose predictive accuracy as the underlying data distribution evolves [AS22]. In many cases, such changes are attributed to unobservable or latent factors, referred to as *hidden contexts*, which are not explicitly represented in the model's features but still influence the target concept [GŽB+14]. In process mining, these context shifts may manifest as structural changes in process behavior [AvZRvdA23].

While various techniques exist for detecting and describing concept drift in process mining (see Adams et al. [AvZRvdA23] for an overview), the present study focuses on identifying *change points* that signal transitions from search to inference behavior in the collected eye-tracking data. This approach involves two main steps: (1) generating a time series from an observed scan-path, and (2) applying a change point detection algorithm.

Time series generation. A common method for deriving time series from traces involves dividing them into fixed-length subtraces [BvdAŽP11, AvZRvdA23]. A so-called *two sliding window* technique is then used to compute pairwise similarity scores between consecutive subtraces (i.e., windows). When these similarity scores are ordered according to the timely occurrence of the subtraces, they form a time series. The characteristics of the resulting time series depend on the chosen similarity measure (cf. Sect. 4) and the length of the subtraces. Although longer subtraces can capture more complex behavioral patterns than

shorter ones, they also require increased computational effort and may overlook structural differences between shorter subtraces.

Change point detection. Once the time series is constructed, change point detection algorithms can be applied. Two primary approaches are commonly used in process mining [AvZRvdA23]:

- Hypothesis testing: compares the distributions of values in adjacent time windows and is applied iteratively across all time points. A significant difference between the distributions indicates a change point.
- Global cost function segmentation: partitions the time series into segments by minimizing a cost function. This function typically includes the sum of deviations from the segment mean and a penalty term for the number of segments, thereby preventing overfitting. The segmentation that minimizes the total cost identifies the most likely change points, corresponding to concept drifts.

6.2 Related Work

This section presents the related work on studies investigating process model comprehension (cf. Sect. 6.2.1), information processing (cf. Sect. 6.2.2), and concept drift analysis to investigate user behavior 6.2.3.

6.2.1 Process Model Comprehension from a Task Perspective

Numerous factors have been identified in research that influence the comprehension of business process models [Fig17, WCI+22]. Among these, task type has received increasing attention due to its significant role in shaping cognitive demands during model interpretation.

According to cognitive fit theory [Ves91], effective problem-solving is facilitated when there is a strong alignment between the nature of

the task and the structure of the problem representation (e.g., a process model). This theory has guided several empirical investigations into how different task types interact with various representational formats. For example, Vessey and Galletta [VG91] distinguished between symbolic tasks, which focus on specific data elements, and spatial tasks, which emphasize relationships among elements. Their study examined how these tasks align with tabular versus graphical representations. Similarly, Ritchi et al. [RJMR19] differentiated between schema tasks, which rely solely on the process model, and non-schema tasks, which require external knowledge, and explored how textual and graphical formats support these tasks.

Dunn and Grabski [DG01] introduced the distinction between *local tasks*, which require attention to a limited area, and *global tasks*, which necessitate integration across multiple areas. Their study demonstrated that the effectiveness of different representational formats (e.g., ER diagrams, tables, text) depends on the spatial distribution of task-relevant information. In the context of fragmented process models, this suggests that task locality plays a critical role in determining cognitive fit. Local tasks, which are confined to a single fragment, are likely to benefit from the abstraction provided by modularization, thereby enhancing cognitive fit. In contrast, global tasks, which require users to integrate information across multiple fragments, may suffer from fragmentation effects, leading to reduced cognitive fit, increased cognitive load, and diminished comprehension performance.

The relevance of task locality in the context of modularization was initially proposed by Reijers and Mendling [RM08] to explain inconsistent findings regarding the benefits of modularization. Subsequent work by Zugal [Zug13] explored this relationship within vertically modularized models, using a limited set of cognitive load measures and without the use of eye-tracking or fine-grained analysis. More recently, Abbad-

Andaloussi et al. [AASS+20] proposed a multi-modal framework for assessing cognitive load in modularized declarative process models.

Building on this theoretical foundation and prior empirical insights, the eye-tracking study further investigates the influence of task locality on horizontal modularization, employing a comprehensive set of cognitive load measures introduced in Section 6.1.3.

6.2.2 Information Processing in Process Model Comprehension

User behavior has been extensively studied in the context of process model comprehension [BSP19, PM13, KHH00, WNP+23, SMN+10, HBZ+16, AAZB+21]. For example, using eye-tracking, Petrusel and Mendling [PM13] demonstrated that task-relevant activities in BPMN models receive significantly more visual attention than task-irrelevant activities. Haisjackl et al. [HBZ+16], using think-aloud protocols, identified two common reading strategies for declarative models in Declare [PSvdA07]: one based on the execution order of activities and another guided by the model's layout orientation. Similarly, in an eye-tracking study of declarative models, Abbad-Andaloussi et al. [AAZB+21] observed that users tend to adopt either exploratory or goal-oriented reading strategies.

While these studies focused on user behavior during comprehension, they did not explicitly identify distinct behavioral phases. This gap was addressed in earlier work by Kim et al. [KHH00], who found that users engaging with multiple interlinked diagrams typically progress through two phases: an initial search phase, followed by hypothesis generation. These were termed perceptual and conceptual integration phases, respectively. However, their analysis was based solely on verbal data. Shanks et al. [SMN+10] also used verbal protocols to identify five behavioral categories during comprehension: preparing, identifying, understanding, articulating, and evaluating. The most frequent

sequence observed began with task understanding, followed by identifying relevant model areas and preparing a solution before final evaluation.

Only a limited number of studies have used eye-tracking to detect specific behavioral phases during process model comprehension. Bera et al. [BSP19] identified two key cognitive processes as critical to comprehension performance: attention and cognitive integration. Attention was measured using fixation count and duration, while cognitive integration was inferred through a combination of eye-tracking and verbal data. Wang et al. [WCI+22] further demonstrated that the integration of process model information can be supported by embedding external information (e.g., rule linking, textual annotations, or diagrammatic cues) directly into the model.

More recently, Winter et al. [WNP+23] investigated gaze patterns during comprehension tasks using models of varying complexity and representational mappings. Through heat maps, focus maps, and scanpath analyses, they identified three recurring gaze patterns: orientation, comprehension, and congruence.

The eye-tracking study builds on this line of research by focusing specifically on the identification of search and inference phases in users' eye-tracking data, contributing to a more fine-grained understanding of cognitive processing during process model comprehension.

6.2.3 Concept Drift Analysis to Investigate User Behavior

In the field of process mining, concept drift analysis has traditionally been applied to business processes [BvdAŽP11, SDFBS21, AvZRvdA23]. These analyses typically begin with an event log composed of multiple traces, from which features are extracted to generate time series

data. However, this approach is not directly applicable to the analysis of user behavior based on individual scan-paths, as it relies on aggregating patterns across multiple instances.

Despite this limitation, several structural features commonly used in process mining for concept drift detection can be adapted for use in the eye-tracking context. These include n-grams (e.g., single activities and directly follows relations), as well as the frequency of activities and their sequential relationships [BvdAŽP11, dSNF+24]. More advanced structural comparisons incorporate loop constructs, choices, and concurrency [OMLR+16, AvZRvdA23]. By applying the graph-based similarity measures proposed in Sect. 4, loop constructs and choices are equally considered for analyzing concept drift in scan-paths. However, concurrency is excluded due to the inherently sequential nature of the scan-paths.

Beyond process mining, concept drift analysis has been widely applied in various domains including user behavior analysis [LLD+18]. Notable applications include recommender systems and text classification (e.g., sentiment analysis and spam detection) [GŽB+14], clickstream analysis for detecting changes in student behavior [PDR+17], and user profiling based on command sequences [IALS11].

6.3 Research Method

To examine the cognitive effects of abstraction and fragmentation, an eye-tracking experiment was conducted in accordance with the empirical standard guidelines for experiments [1]. Sections 6.3.1 through 6.3.3 outline the study design, execution procedures, and data analysis methods.

[1] https://github.com/acmsigsoft/EmpiricalStandards (Accessed: 18.07.2025)

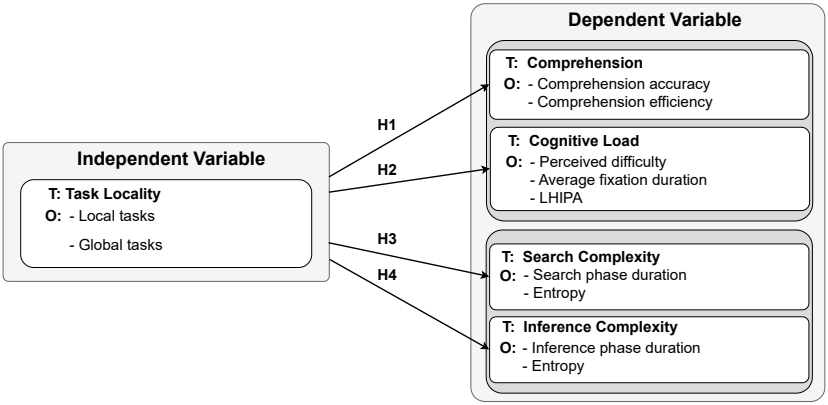


Figure 6.4: Research model for hypothesis testing. T - Theoretical construct, O - Operationalization of construct.

6.3.1 Study Design

Research Model

Building on the theoretical foundations outlined in Sect. 6.1, the present empirical study investigates the effects of abstraction and fragmentation in modularized process models on user performance and cognitive processing. Specifically, the study examines their impact on comprehension and cognitive load at the task level (RQ1), as well as on search and inference complexity at the phase level (RQ2). The underlying research model is illustrated in Figure 6.4.

The study manipulates task locality as the independent variable, operationalized through two levels: *local tasks*, which involve control-flow reasoning within a single process fragment, and *global tasks*, which require integration of control-flow information across two separate fragments. As discussed in Section 6.1.1, both task types benefit from abstraction due to modularization. However, only global tasks are neg-

atively affected by fragmentation, as they necessitate cross-fragment integration. The dependent variables include: comprehension, cognitive load, search complexity, inference complexity. These constructs are operationalized by applying the measures introduced in Sect. 6.1.3 and Sect. 6.1.4. The relationships among these variables are depicted in the research model (Fig. 6.4).

Concerning comprehension and cognitive load (*RQ1*), the following hypotheses are formulated:
◦ $\mathbf{H}_1$: Global tasks yield lower task comprehension than local tasks.
◦ $\mathbf{H}_2$: Global tasks yield higher cognitive load than local tasks.

Concerning search and inference phases (*RQ2*), the following hypotheses are formulated:
◦ $\mathbf{H}_3$: Global tasks yield more complex search behavior than local tasks.
◦ $\mathbf{H}_4$: Global tasks yield more complex inference behavior than local tasks.

Material

Following the fragment-based modeling approach proposed by Hewelt and Weske [HW16], the experimental setup consists of a set of process model fragments representing a logistics scenario and a corresponding set of comprehension tasks (see Appendix D). These tasks require participants to evaluate control-flow relationships between activities within the process model. Depending on the scope of the information required, tasks are categorized as either local, involving a single process fragment, or global, requiring integration across two distinct fragments.

The logistics process is decomposed into six interconnected fragments. These are linked via three data objects, whose state transitions are modeled using dedicated lifecycle diagrams. The design of the fragments adheres to established process modeling guidelines [MRvdA10, BRVU02], ensuring a consistent layout and manageable complexity. Each fragment contains between 6 and 8 activities and 2 to 4 gateways (exclusive or parallel). To minimize the influence of domain-specific knowledge, activity labels were formulated using lay terminology. In total, the model comprises 41 activities and 46 data-object interactions, and incorporates commonly used BPMN constructs, including start and end events, tasks, sequence flows, parallel gateways, and data-based XOR gateways [zMR13]. This results in higher model complexity compared to prior studies [Zug13, RMD11].

The experiment includes eight comprehension tasks designed to assess participants' understanding of control-flow relationships, which encompass: sequence, choice, repetition, and concurrency between activities. These relationships are either confined to a single fragment (local tasks) or span across two fragments (global tasks), thereby covering a range of workflow patterns reflective of real-world comprehension scenarios.

Each task is presented as a statement, following a standardized template (see Fig. 6.5), which participants are instructed to evaluate as either correct or incorrect based on the semantics of the process model. The task formulation ensures that relevant information is clearly identifiable within the model. Participants are explicitly instructed to continue searching until all task-relevant information has been located, encouraging the use of the mental list stopping rule [BPW07]. In this context, the search phase is defined as the period from task onset until both task-relevant activities are identified, while the inference phase extends from that point until task completion (see Fig. 6.3). These de-

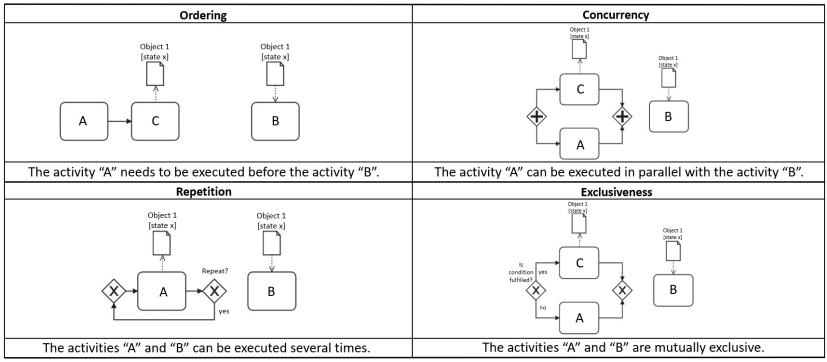


Figure 6.5: Four different control-flow patterns and how they are modeled for global tasks based on the fragment-based modeling approach.

sign choices support the clear separation of behavioral phases in the eye-tracking data, as further detailed in Sect. 6.3.2 and Sect. 6.3.3. To mitigate potential learning effects, each task targets distinct aspects of the process model fragments.

The experiment was conducted using the EyeMind framework for data collection and empirical investigation [AALW23]. The framework particularly supports the precise mapping of users' eye fixations to corresponding elements within a process model. A screenshot of the EyeMind user interface used during the experiment is shown in Figure 6.6. Participants interacted with the interface to complete comprehension tasks. The process model fragments and their corresponding lifecycle diagrams were provided as separate files, accessible via the file explorer located on the left side of the interface. Crucially, only one file could be viewed at a time, requiring participants to switch between files as needed. The task prompt was displayed at the top of the screen, ensuring it remained visible throughout the task execution.

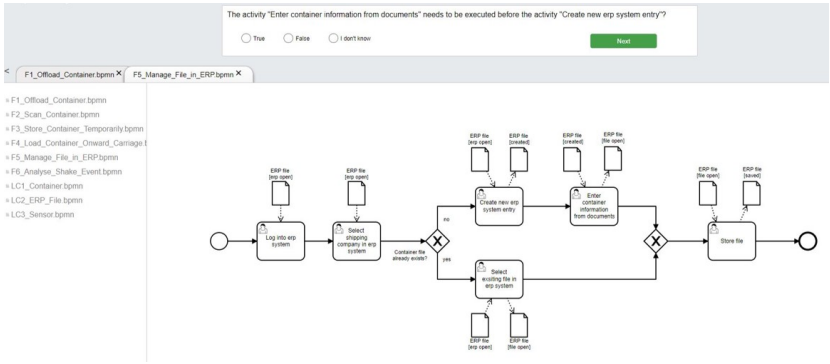


Figure 6.6: User interface with an example of an experiment question.

Participants

Table 6.1 presents a summary of the demographic characteristics of the study participants. A total of 46 individuals were recruited for the experiment. Among them, 22 participants were affiliated with the University of St. Gallen, 17 with the Karlsruhe Institute of Technology, 4 with the Forschungszentrum für Informatik (FZI) in Karlsruhe, and 3 with Promatis, an IT company. Participants ranged in age from 20 to 50 years, with 63% falling within the 20-30 age bracket. The sample included individuals from diverse professional backgrounds: 22 were engaged in academic research, 17 were students at various stages of undergraduate and graduate education, and 7 were employed in the IT industry.

Regarding familiarity with Business Process Model and Notation (BPMN), 48% of participants reported a high level of familiarity (scores between 5 and 7 on a 7-point scale, where 1 indicates unfamiliarity and 7 indicates high familiarity), while 42% indicated low familiarity (scores between 1 and 3). To ensure consistent understanding of the experimental material, all participants underwent a standardized familiar-

Demographics	Groups	Count	Percentage (%)
Education	Business Informatics	15	0.33
	Cognitive Science	1	0.02
	Computer Science	25	0.54
	Industrial Engineering	4	0.09
	Mechatronics	1	0.02
Profession	Research	22	0.48
	Student	17	0.37
	Industry	5	0.11
	IT-Admin	2	0.04
Experience with BPM	0 month	18	0.39
	$\leq$ 1 year	12	0.26
	$\leq$ 2 years	2	0.04
	$\leq$ 5 years	7	0.15
	$>$ 5 years	7	0.15
Familiarity with BPMN	1 (strongly disagree)	9	0.2
	2	7	0.15
	3	3	0.07
	4 (neutral)	5	0.11
	5	8	0.17
	6	9	0.2
	7 (strongly agree)	5	0.11

Table 6.1: Demographic data of the participants.

ization session covering the BPMN concepts and the interpretation of process models developed using the fragment-based approach. Following this session, participants completed a set of test tasks analogous to those used in the main experiment to assess their comprehension and readiness.

6.3.2 Experiment Procedure

The experiment was conducted through individual eye-tracking sessions, each lasting approximately one hour. As illustrated in Fig. 6.7, each session began with a familiarization phase in which participants were introduced to the BPMN concepts embedded in the model fragments, as well as key principles of the fragment-based modeling approach [HW16]. Following this introduction, participants completed an oral quiz designed to assess their understanding of the modeling approach and to clarify any BPMN concepts that posed difficulties. All participants successfully completed the quiz, demonstrating sufficient comprehension to proceed with the experiment. Clarifying questions were permitted during this phase to ensure conceptual accuracy.

Subsequently, participants completed screening and demographic questionnaires to confirm their eligibility for eye-tracking (e.g., failing eyesight) and to collect background information such as gender, age range, and familiarity with the relevant concepts. Prior to data collection, participants were seated in front of the eye-tracking device and received instructions regarding the procedure. Calibration of the device was then performed to ensure accurate gaze tracking. Participants were instructed to minimize head movements during the session.

To mitigate the development of a holistic understanding of the entire process model, comprising six fragments and three life cycles, participants were explicitly instructed to treat each task independently. They were encouraged to focus on identifying relevant activities within each task and to infer their interrelations only after completing the search phase. This approach was intended to promote the use of the mental list stopping rule during the search process (cf. Sect. 6.1.4). Accordingly, participants were expected to first locate all relevant activi-

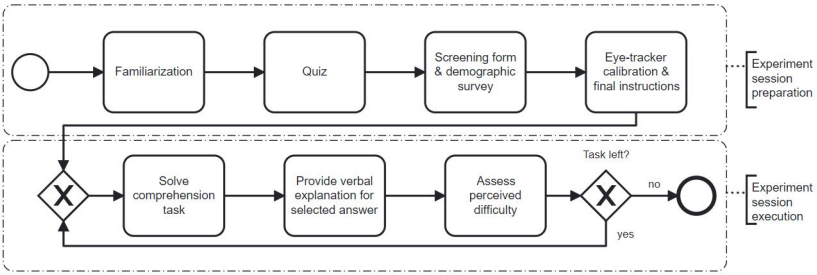


Figure 6.7: The different steps during each experiment session.

ties and subsequently deduce their relationships during the inference phase.

During the data collection phase, participants completed a series of tasks presented in randomized order. After each task, they were asked to justify their responses and complete a self-assessment questionnaire evaluating perceived task difficulty. Subsequently, the eye-tracking data recorded during task execution, the task solutions, and the self-assessment responses were used for the analysis.

6.3.3 Data Analysis

The data analysis encompasses the investigation of (*RQ1*) differences in users' comprehension and cognitive load and (*RQ2*) differences in users' search and inference behavior. Furthermore, the analysis investigates (*RQ3*) the detection of concept drifts, indicating a transition from search to inference behavior.

Comprehension and Cognitive Load

The comprehension and cognitive load measures described in Sect. 6.1.3 were computed at the task level. Each of the 46 participants completed four tasks per factor level (i.e., local and global tasks), resulting in 184

data points per condition. To mitigate intra-individual dependencies, the mean value across the four tasks for each factor level was calculated per participant, yielding 46 paired data samples.

Due to technical issues with the eye-tracking equipment, the dataset was further reduced. Specifically, fixation-based measures were available for 44 participants, while pupil-based measures were available for 43 participants. In two cases, excessive ambient brightness interfered with accurate gaze detection. In one case, frequent head movements compromised the reliability of pupil dilation measurements.

The remaining data were used to compute both descriptive and inferential statistics to evaluate the study hypotheses (cf. Table 6.2). For inferential analysis, the non-parametric Wilcoxon Signed-Rank test (single-tailed) [Con99] was employed, as it is appropriate for comparing paired samples without assuming normal distribution. Additionally, effect size estimates (r^2) were calculated following the method proposed by Fritz et al. [FMR12]. The r^2 statistic quantifies the proportion of variance in the dependent variable that can be attributed to the independent variable, in this case, the locality of the task.

Search and Inference Behavior

The comparison of search and inference behavior was conducted in two stages. First, the proposed phase segmentation method was validated to ensure reliable differentiation between the search and inference phases within the experimental tasks. Second, a statistical analysis was performed to examine whether these two phases exhibit distinct cognitive characteristics when comparing local and global task conditions.

Phase separation and validation. The methodology used to infer participants' behavior during the search and inference phases is summa-

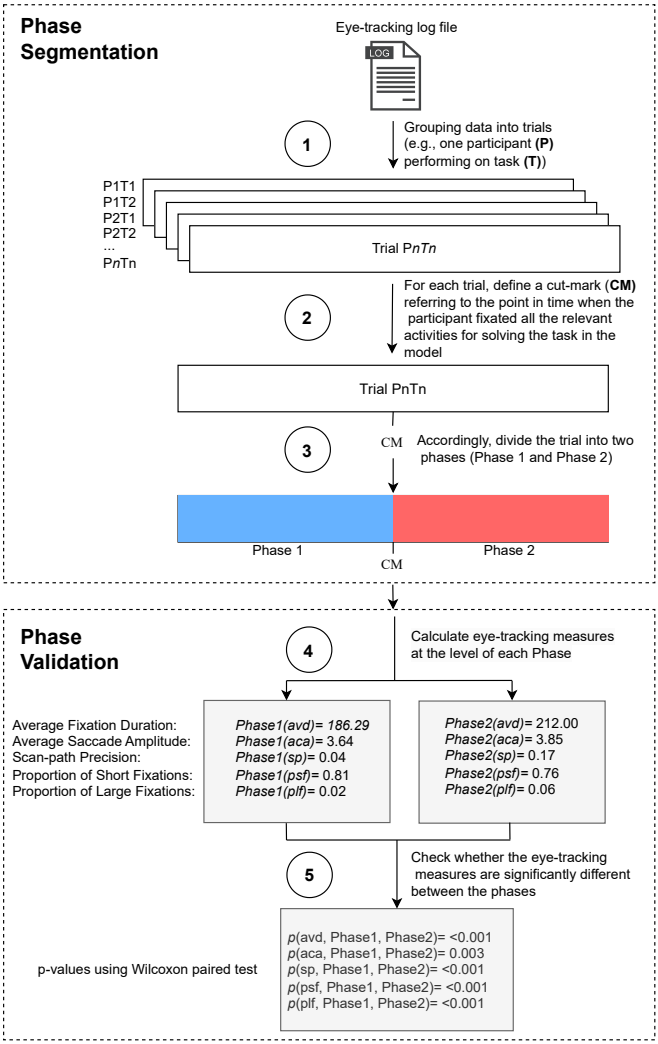


Figure 6.8: Approach for the segmentation and validation of the search and inference phases.

rized in Fig. 6.8. This figure outlines the segmentation of eye-tracking data into two distinct cognitive phases, followed by a validation of the segmentation procedure.

1. Trial Structuring: Eye-tracking data collected from participants were first organized into individual trials. Each trial corresponds to a single participant performing a specific task (e.g., Participant P1 performing Task T1, Participant P2 performing Task T1).
2. Phase Segmentation: A cut-mark was assigned to each trial at the time point when the participant fixated on the second of the two task-relevant activities. This point marks the transition from the search phase to the inference phase (cf. Fig. 6.3).
3. Phase Definition: Based on the cut-mark, each trial was divided into two segments: Phase 1 (preceding the cut-mark) and Phase 2 (following the cut-mark). Fig. 6.3 illustrates this segmentation using fixation data from a representative local task.
4. Cognitive Measure Analysis: To assess whether the two phases reflect distinct cognitive processes, several eye-tracking metrics associated with search and inference behavior (as introduced in Sect. 6.1.4) were computed separately for each phase. As shown in Fig. 6.8, Phase 1 was characterized by shorter average fixation durations, smaller saccade amplitudes, lower scan-path precision, a higher proportion of short fixations (< 250ms), and a lower proportion of long fixations (≥ 500ms) compared to Phase 2.
5. Statistical Validation: The Wilcoxon Signed-Rank test (single-tailed), suitable for paired comparisons without assuming normality, was applied to evaluate the statistical significance of these differences. All observed differences between Phase 1 and Phase 2 were statistically significant.

Taken together, the observed trends in eye-tracking measures, the theoretical framework outlined in Sect. 6.1.4, and the experimental instructions provided to participants (cf. Sect. 6.3.2) support the interpretation that Phase 1 predominantly reflects search behavior, while Phase 2 corresponds to inference behavior.

Hypotheses Testing

To investigate differences in search and inference behavior, the visual behavior measures introduced in Sect. 6.1.3 were computed at the phase level. The data selection procedure, and the calculated descriptive and inferential statistics are consistent with the analysis of comprehension and cognitive load at task level. This resulted in 44 paired data samples for estimating phase duration and entropy across local and global task conditions. Only two trials were excluded due to inaccurate gaze detection.

Concept Drift Detection

Building on the analysis of search and inference behavior, the concept drift analysis aims to identify change points that signal transitions between these two phases. The focus of this analysis lies in the generation of time series from scan-path data, using the similarity measures introduced in Chapter 4.

Time series generation. To construct time series from scan-paths, a sliding window approach is applied, as described in Sect. 6.1.5. Based on this approach, each scan-path is segmented into subtraces of equal length. For each pair of consecutive subtraces, a similarity score is computed. To assess the suitability of the similarity measures from Chapter 4 for capturing temporal structural variations in scan-paths, a separate time series is generated for each measure. Only those measures are applied from Chapter 4 that showed a relatively high perfor-

mance with respect to the considered eye-tracking data sets (EyeT1 and EyeT2) in comparison to the other similarity measures, i.e., all measures based on cosine distance (C1, C2, C3, Cg Agg(C1,C2), Agg(C1,C2,Cg)) and based on Jaccard distance (J1, J2, J3, Jg, Agg(J1,J2), Agg(J1,J2,Jg). Additionally, Levenshtein distance (LD) is considered as a baseline.

A critical parameter in the sliding window approach is the window size, which influences both sensitivity and computational efficiency. Consequently, multiple window sizes are evaluated. Given the average scan-path lengths, i.e., 408 fixations for global tasks and 183 for local tasks, different window sizes are selected for each task type. For local tasks, window sizes of 20, 30, 40, 50, and 60 fixations are used; for global tasks, window sizes of 20, 40, 60, 80, and 100 are used.

Concept drift detection algorithm. To detect change points within the generated time series, the binary segmentation algorithm [TOV20] is employed. This method was selected based on the comprehensive review by Truong et al. [TOV20], which highlights its effectiveness in offline change point detection and its ability to constrain the number of detected changes to one. The implementation used is provided by the Python package ruptures [TOV20]. Alternative algorithms available in the same package are also tested, but no significant performance differences are observed when applied to the eye-tracking data [Aki24].

Evaluation. The evaluation of change point detection performance is based on two commonly applied measures in concept drift research [AC17], which quantify the temporal deviation between actual change point $\text{acp}(\sigma_i)$ and the detected change point $\text{dcp}(\sigma_i)$ for σ_i:

- *Mean Absolute Error (MAE)* [AC17]:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |\text{acp}(\sigma_i) - \text{dcp}(\sigma_i)|$$

MAE measures the average magnitude of error over all traces $n = |L|$, providing an intuitive indication of prediction accuracy. However, it does not account for the direction of the error.

- *Mean Signed Difference (MSD)* [AC17]:

$$\text{MSD} = \frac{1}{n} \sum_{i=1}^{n} (\text{acp}(\sigma_i) - \text{dcp}(\sigma_i))$$

MSD captures the directional bias of the predictions, indicating whether change points tend to be detected earlier or later than the actual transition. However, it may obscure large errors if they cancel each other out.

6.4 Findings

The following sections present the findings regarding comprehension and cognitive load (*RQ1*), search and inference behavior (*RQ2*), and concept drift analysis (*RQ3*).

6.4.1 Comprehension and Cognitive Load

As shown by the descriptive statistics in Table 6.2, *comprehension accuracy* (scored on a scale from 0 = incorrect to 1 = correct) was significantly lower for global tasks compared to local tasks, as indicated by the corresponding $p-$value. Similarly, *comprehension efficiency* (measured in seconds) was significantly reduced in global tasks, suggesting that participants required more time to arrive at a solution.

Regarding cognitive load, participants reported significantly higher *perceived difficulty* for global tasks (on a scale from 0 = “very easy" to 4 = “very difficult"). In addition, *average fixation duration* (in milliseconds), an objective indicator of cognitive effort, was significantly

Hypothesis/Construct	Measure	Descriptive			Inferential	
		N	Local M (SD)	Global M (SD)	p-value	r^2
H_1/Comprehension	Accuracy	46	0.978 (0.071)	0.739 (0.247)	$< .001$	0.575
	Efficiency	46	54.097 (16.823)	117.593 (41.5)	$< .001$	0.730
H_2/Cognitive load	Perceived difficulty	46	0.62 (0.521)	1.891 (0.772)	$< .001$	0.754
	Avg. fix. duration	44	194.323 (25.526)	200.966 (25.421)	$< .001$	0.290
	LHIPA	43	1.185 (0.298)	0.815 (0.216)	$< .001$	0.668

Table 6.2: Descriptive and inferential statistics showing the impact of the task locality on users' performance and cognitive load. N: number of observations, M: calculated mean, SD: calculated standard deviation. Units: Phase duration in seconds, Average fixation duration in milliseconds. A *p-value*$< .05$ means that the pairwise comparison results are significant, r^2 thereby indicates the effect size of the independent variable.

greater during global tasks. Furthermore, the *LHIPA score* was significantly lower for global tasks, indicating higher cognitive effort in comparison to local tasks.

In line with the theoretical framework outlined in Sect. 6.1.3, these findings support hypotheses H_1 and H_2: global tasks are associated with lower comprehension (H_1) and higher cognitive load (H_2) compared to local tasks. This indicates a significant effect of task locality on both comprehension and cognitive load at the task level.

6.4.2 Search and Inference Behavior

Table 6.3 presents the results of the comparative analysis of search and inference behavior between local and global tasks. During the *search phase*, the *phase duration* (measured in seconds) was significantly longer for global tasks compared to local tasks. Additionally, *scan-path variability*, as measured by entropy, was higher in global tasks, indicating more dispersed visual exploration.

Hypothesis/Construct	Measure	Descriptive			Inferential	
		N	Local M (SD)	Global M (SD)	p-value	r^2
H_3/Search	Search phase duration	44	28.075 (21.918)	45.084 (24.983)	$< .001$	0.607
	Entropy	44	3.94 (0.686)	4.65 (0.473)	$< .001$	0.567
H_4/Inference	Inference phase duration	44	24.856 (19.636)	77.837 (59.236)	$< .001$	0.755
	Entropy	44	3.154 (0.446)	4.709 (0.626)	$< .001$	0.731

Table 6.3: Descriptive and inferential statistics showing how users' search and inference phases differ between local and global tasks in terms of phase duration, cognitive load and scan-path variability. N: number of observations (cf. Sect. 6.3.3), M: calculated mean, SD: calculated standard deviation. Units: Phase duration in seconds, Average fixation duration in milliseconds. A *p-value*$< .05$ means that the pairwise comparison results are significant, r^2 thereby indicates the effect size of the independent variable.

In the *inference phase*, a similar pattern was observed: both *phase duration* and *entropy* values were significantly higher for global tasks than for local tasks, suggesting increased cognitive effort and complexity in processing.

These findings align with the theoretical framework outlined in Sect. 6.1.3 and provide empirical support for hypotheses H_3 and H_4. Specifically, global tasks are associated with more complex search behavior H_3 and more complex inference behavior H_4, indicating a significant effect of task locality on cognitive processing at the phase level. These results are further elaborated in Sect. 6.5.

6.4.3 Concept Drift Detection

The evaluation results for the various similarity measures applied in concept drift detection are presented in Table 6.4 (based on MAE) and Table 6.5 (based on MSD).

Analysis of the MAE values indicates that the graph-based similarity measures Agg(C1,C2,Cg) and Agg(J1,J2,Jg) perform particularly well

for larger window sizes (i.e., for global tasks with w $\geq$ 60). In these cases, they consistently outperform their non-graph-based aggregated measures Agg(C1,C2) and Agg(J1,J2). Furthermore, Agg(J1,J2,Jg) achieves the best performance across all evaluated similarity measures for larger window sizes, i.e., for w=60 and 80 in global tasks and w=50 and 60 in local tasks. Overall, similarity measures based on Jaccard distance (J1, J2, Agg(J1,J2), and Agg(J1,J2,Jg)) demonstrate robust performance across both global and local task settings.

Despite these findings, the MAE values remain relatively high when compared to the average scan-path lengths. However, performance improves across all distance measures as the window size increases.

The MSD results reveal a consistent trend across all similarity measures: the average detected change point tends to precede the actual change point defined in Section 6.3.3. Interestingly, the graph-based measure Cg exhibits strong performance in terms of MSD for window sizes w = 20 and w = 40, a result attributed to the high standard deviation of the computed values. A similar pattern is observed for the Levenshtein distance (LD) in global tasks with a window size w=80.

Among all evaluated measures, the graph-based measure Agg(J1,J2,Jg) outperforms the others for larger window sizes in terms of both MAE and MSD.

6.5 Discussion

This section discusses the key findings and limitations related to (1) the eye-tracking study on process model comprehension tasks and (2) the subsequent analysis of structural process variety for detecting concept drifts in user behavior.

The results presented in Section 6.4.1 indicate that task type significantly affects user comprehension and cognitive load when interact-

ing with horizontally modularized process models. These findings support existing hypotheses in the literature [Zug13, RM08], which suggest that global tasks impose higher cognitive demands than local tasks, which benefit from abstraction. Given that horizontal modularization allows for flexible decomposition of process models without strict hierarchical or cross-cutting constraints, it is reasonable to infer that these findings may extend to other modularization strategies, even those with more rigid decomposition rules.

The insights from Section 6.4.2 reveal notable differences in phase duration and scan-path entropy between local and global tasks. In particular, inference phases in global tasks exhibit increased complexity, likely due to fragmentation. This is reflected in higher entropy values, representing greater uncertainty in fixation patterns, during inference phases compared to search phases in global tasks. In contrast, local tasks show the opposite trend. Additionally, the duration of inference phases in global tasks is approximately three times longer than in local tasks.

Furthermore, the concept drift analysis in Section 6.4.3 provides initial validation that the incorporation of structural process variety enhances the detection of concept drifts in users' visual scan paths, across both local and global comprehension tasks. The relevance of structural properties appears to increase with the length of the subtraces used for drift detection. When comparing concept drift detection results across identical window sizes (w = 20, 40, 60), the error rates (measured by MAE) of the detected change points are consistently lower for local tasks than for global tasks, regardless of the applied similarity measure.

These findings have several implications for both future research and practical applications in process model comprehension. From a research perspective, task type and information processing phases should

be considered fundamental dimensions in the analysis of process model comprehension. While the relationship between task characteristics and cognitive processing has been explored in other domains [Vak99, HAC13, CWL+19], it remains underexplored in the context of process modeling. The potential for such research has been highlighted by Mandelburger and Mendling [MM21], who proposed a theoretical cognitive framework for task performance with diagrams. The conducted eye-tracking study contributes initial empirical evidence in support of this direction.

From a practical standpoint, the gained insights may support more accurate assessments of the specific challenges users face during task execution, whether related to locating relevant information, drawing correct inferences, or both. Eye-tracking data in combination with the introduced structural process similarity measures could enable the delivery of personalized, real-time support, such as providing additional guidance when comprehension difficulties are detected at the phase level. This approach could be particularly valuable in e-learning environments [PRP12] focused on teaching modularized models, as well as in professional contexts such as software development [MFF+23] and logistics [BMPR12], where modularized models are frequently used.

Threats to Validity

Several potential threats to the validity of the eye-tracking study must be considered. Although all experimental sessions were conducted under a rigorously defined protocol and within a controlled environment, the influence of confounding variables cannot be entirely excluded. To enhance internal validity, multiple measures were implemented. Task order was randomized across participants to mitigate learning effects, and each comprehension task involved a unique combination of two activities, evenly distributed across six process fragments. This en-

sured that each fragment was relevant to only two of the eight tasks. Additionally, the relatively large sample size (46 participants) strengthens the reliability of the findings.

To ensure consistent understanding of the process models, all participants received a standardized introduction to BPMN and the fragment-based modeling approach. This was followed by a familiarization phase and a quiz to confirm baseline knowledge.

To control for potential confounds in task formulation, all local and global tasks were phrased identically, each referring to exactly two activities and encompassing the same set of control-flow patterns (see Section 6.3.1). Furthermore, all tasks were based on the same modularized process model, composed of six equally complex fragments (see Appendix D). Nonetheless, there remains a possibility that certain fragments introduced unforeseen cognitive complexity, potentially biasing the comparison between task types. This risk is mitigated by the high number of data points collected per factor level (184), as detailed in Section 6.3.1.

In the context of eye-tracking during process model comprehension, the analysis assumes a sequential progression in cognitive processing, i.e., that the search phase concludes once task-relevant activities are identified, and the inference phase begins immediately thereafter, without overlap. This assumption implies a discrete transition between the two phases. However, cognitive theories suggest that mental processes can occur in parallel [WWPP06], which challenges the strict separation of search and inference. As such, the potential for overlapping cognitive activity introduces an inherent uncertainty that must be acknowledged in the interpretation of the results. Nonetheless, this risk is mitigated by the task design, which included an implicit stopping rule: participants were instructed to first locate the two specified activities and then infer their relationship. Statistical validation

of the segmentation, presented in Section 6.3.3, supports the distinction between phases. Metrics such as average fixation duration, saccade amplitude, scan-path precision, and the distribution of short and long fixations all indicate a clear separation between search and inference phases, suggesting that participants adhered to the intended task structure.

Regarding the evaluation of the concept drift detection algorithm, it is important to note that shorter window sizes (subtraces) reduce the temporal resolution of the time series, potentially leading to premature or delayed detection of concept drifts. This limitation may partially explain the observed trend of decreasing MAE values across all similarity measures as window size increases.

Finally, the external validity of the findings warrants consideration. To enhance generalizability across different process modeling languages, the comprehension tasks incorporated a variety of commonly used control-flow patterns (see Figure 6.5). These patterns are widely applicable, regardless of the specific modeling language.

The extent to which the experimental comprehension tasks reflect real-world scenarios is also a relevant concern. Three factors support their practical validity: (1) the control-flow patterns used are prevalent in practice [zMR13], (2) both local and global comprehension tasks are commonly encountered in real-world settings [MFF+23, BMPR12], and (3) the stopping rules embedded in the tasks mirror search strategies observed in other domains [BP04, BPW07]. Nevertheless, in more complex real-world tasks, the distinction between search and inference may not always be as clear-cut. To date, no studies have specifically examined search strategies in the context of process model comprehension.

Participant heterogeneity also introduces potential variability. However, the diverse levels of process modeling experience among partic-

ipants (see Section 6.3.1) suggest that the findings are not overly dependent on prior expertise. All participants had an educational background in information systems and completed a preparatory phase to ensure a uniform understanding of BPMN and the modeling approach. This ensured that all participants were capable of completing the tasks, regardless of their individual backgrounds. The within-subject design of the experiment further enhances robustness by exposing each participant to all treatment conditions.

6.6 Summary

In summary, this chapter demonstrates how structural process variety can enhance the analysis of user behavior using eye-tracking data. By leveraging the structural similarity measures introduced in Chapter 4, the conducted analysis demonstrates improved detection of concept drift in user behavior, indicating a transition from information search to information inference. The findings are based on a controlled eye-tracking study that systematically explores the impact of abstraction and fragmentation in modularized process models. The study design enables a clear distinction between cognitive phases. The resulting insights from the analysis, considering different task types and cognitive phases, offer valuable guidance for developing targeted user support to facilitate process model comprehension.

		Mean Absolute Error																			
	w	LD		C1		C2		J1		J2		Cg		Agg(C1,C2)		Agg(C1,C2,Cg)		Jg		Agg(J1,J2)		Agg(J1,J2,Jg)	
G.	20	125.776	(89.654)	124.019	(100.317)	117.942	(87.993)	120.846	(97.59)	**114.25**	(84.726)	141.192	(106.557)	122.288	(97.815)	129.885	(97.487)	131.147	(92.107)	122.654	(99.347)	135.827	(108.951)
	40	118.808	(95.171)	106.487	(87.567)	112.994	(95.082)	**106.282**	(88.562)	111.115	(94.084)	118.218	(96.836)	107.904	(88.318)	112.654	(93.794)	110.532	(84.433)	108.558	(89.65)	112.603	(95.67)
	60	99.833	(84.951)	93.058	(84.502)	92.769	(79.622)	95.006	(84.558)	95.737	(84.332)	97.891	(79.84)	95.436	(84.641)	92.603	(82.345)	92.897	(77.484)	95.41	(85.059)	**91.788**	(81.542)
	80	85.192	(80.174)	80.801	(74.308)	80.699	(74.008)	81.205	(73.745)	82.904	(75.068)	82.609	(75.89)	81.09	(74.188)	80.699	(73.476)	82.679	(76.16)	82.327	(74.462)	**79.686**	(73.95)
	100	70.083	(68.701)	70.173	(67.929)	73.564	(70.841)	**68.577**	(66.918)	73.66	(71.04)	71.462	(69.285)	71.103	(68.392)	70.782	(68.574)	72.442	(70.404)	**68.577**	(66.918)	70.885	(68.98)
L.	20	71.15	(72.523)	**66.940**	(63.860)	72.24	(72.649)	68.581	(63.729)	70.904	(64.342)	69.533	(47.265)	67.647	(63.845)	69.629	(63.678)	79.916	(78.748)	67.545	(63.731)	71.581	(71.969)
	30	63.802	(70.185)	64.844	(72.207)	64.725	(71.670)	**63.383**	(71.195)	65.006	(72.223)	67.479	(69.094)	64.533	(71.661)	65.024	(71.19)	65.096	(49.177)	63.509	(71.157)	64.024	(70.33)
	40	58.275	(70.194)	57.257	(70.58)	57.563	(70.478)	**55.964**	(69.575)	57.533	(70.489)	58.539	(70.001)	57.257	(70.58)	57.419	(70.39)	59.527	(70.664)	57.228	(70.589)	56.287	(68.959)
	50	52.317	(70.065)	51.934	(70.165)	51.868	(70.218)	51.814	(70.079)	51.898	(70.203)	51.928	(68.342)	51.904	(70.178)	51.695	(69.772)	52.317	(69.506)	51.904	(70.188)	**50.455**	(68.217)
	60	46.234	(67.902)	46.263	(67.788)	46.144	(67.847)	46.204	(67.813)	46.198	(67.924)	46.048	(66.822)	46.174	(67.828)	46.234	(67.662)	46.198	(67.476)	46.174	(67.828)	**45.539**	(66.585)

Table 6.4: Evaluation of the concept drift detection based on MAE for different similarity measures and window sizes (w). The values are separately calculated for global tasks (G.) and local tasks (L.). The values in brackets indicate the standard deviations for the calculated mean values.

		Mean Signed Difference																			
	w	LD		C1		C2		J1		J2		Cg		Agg(C1,C2)		Agg(C1,C2,Cg)		Jg		Agg(J1,J2)		Agg(J1,J2,Jg)	
G.	20	-76.609	(134.36)	-62.186	(147.144)	-86.288	(119.37)	-69.526	(139.129)	**-86.994**	(112.69)	**-50.167**	(169.957)	-66.353	(142.083)	-69.077	(147.242)	-79.622	(139.333)	-71.577	(140.906)	-73.212	(158.252)
	40	-79.526	(129.995)	-92.603	(102.226)	-96.288	(112.068)	**-96.000**	(99.682)	-92.795	(112.301)	**-85.872**	(126.576)	-95.199	(101.967)	-95.551	(111.27)	-102.635	(93.932)	-93.34	(105.496)	-87.026	(119.547)
	60	-85.808	(99.183)	-79.173	(97.711)	-83.179	(89.654)	-80.071	(98.904)	-82.378	(97.502)	-93.532	(84.937)	-81.641	(98.095)	-81.192	(93.683)	-88.5	(82.503)	-80.423	(99.435)	**-77.506**	(95.303)
	80	**-65.103**	(97.297)	-70.199	(84.457)	-68.788	(85.259)	-72.026	(82.789)	-71.481	(86.082)	-75.455	(83.05)	-70.038	(84.763)	-69.878	(83.897)	-72.282	(86.15)	-70.904	(85.477)	**-67.506**	(85.282)
	100	-51.929	(83.362)	-53.981	(81.472)	-56.577	(85.108)	**-51.962**	(80.584)	-56.096	(85.676)	-56.641	(81.922)	-53.051	(83.265)	-54.269	(82.345)	-55.904	(84.221)	**-51.962**	(80.584)	**-51.769**	(84.368)
L.	20	-64.599	(78.449)	-61.754	(68.916)	-66.994	(77.541)	-62.054	(70.137)	-64.928	(70.403)	-60.347	(58.602)	-62.473	(68.945)	-64.599	(68.805)	-72.084	(86.017)	**-59.868**	(71.033)	-65.257	(77.782)
	30	-60.018	(73.467)	-61.934	(74.733)	-64.329	(72.027)	**-59.659**	(74.363)	-62.024	(74.815)	-62.473	(73.678)	-64.21	(71.953)	-64.689	(71.496)	-64.868	(49.478)	-59.689	(74.41)	-62.802	(71.429)
	40	-57.473	(70.857)	-56.335	(71.323)	-56.725	(71.159)	**-55.018**	(70.33)	-56.754	(71.121)	-57.772	(70.639)	-56.335	(71.323)	-56.545	(71.099)	-58.76	(71.306)	-56.365	(71.283)	-55.437	(69.648)
	50	-50.198	(71.608)	-49.659	(71.803)	-49.749	(71.744)	-49.599	(71.673)	-49.778	(71.731)	-49.808	(69.911)	-49.689	(71.774)	-49.24	(71.537)	-50.198	(71.061)	-49.689	(71.783)	**-48.251**	(69.802)
	60	-42.353	(70.402)	-42.204	(70.402)	-42.323	(70.308)	-42.263	(70.351)	-42.413	(70.363)	-42.263	(69.292)	-42.293	(70.328)	-42.114	(70.316)	-42.413	(69.931)	-42.293	(70.328)	**-41.754**	(69.036)

Table 6.5: Evaluation of the concept drift detection based on MSD for different similarity measures and window sizes (w). The values are seperately calculated for global tasks (G.) and local tasks (L.). The values in brackets indicate the standard deviations for the calculated mean values.

7 Conclusion and Outlook

This chapter summarizes the research contributions of this work and addresses its main limitations. Finally, an outlook is provided, proposing potential directions for future research.

7.1 Contributions

Overall, the contributions to the research on process analysis in the context of BPM and process mining can be summarized as follows.

- *Identification of formal properties for the measurement of structural process variety*: The study of the literature on diversity measurement revealed key properties essential for quantifying structural process variety. Specifically, a meaningful assessment of structural variety requires a metric that captures the pairwise distances between process traces. To ensure uniqueness, the distance metric used must satisfy the properties of an ultrametric, which inherently reflects the hierarchical relationships among traces.
- *Proposition of a measurement framework for structural process variety*: The proposed measurement framework enables the calculation of structural process variety through pairwise trace comparisons, thereby ensuring the preservation of the ultrametric property. This is achieved by applying hierarchical clustering of the traces. The framework is evaluated based on a set of event logs, which contain a diverse set of structural properties. The frame-

work yields better outcomes relative to other structural variety measures proposed in the literature, due to the manner in which it captures the structural deviations between traces.

- *Identification of formal properties for the measurement of structural trace similarity*: To further improve the measurement of structural process variety, ten properties are identified, which a similarity measure should fulfill. To better account for the structural similarity the notion of an edge type is introduced, indicating whether certain structural patterns (loops, skips, and choices) are present in a trace. The identified properties show that existing trace similarity measures in the literature do not well reflect structural properties of the traces.
- *Proposition of a novel measurement approach for structural trace similarity*: Based on the observed limitations in existing similarity measures, a novel graph-based measurement approach is proposed. The proposed measurement approach, based on graph analysis, allows to include edge types into the measurement of trace similarity. The approach entails the transformation of the traces into graph representations and the subsequent application of graph algorithms to derive structural trace features. Utilizing vector-space embedding and set comparison, these features allow to measure structural similarity between traces independently from the respective sets of activities. The approach is evaluated using several synthetic and real-life event logs. The evaluation shows that the measurement approach outperforms existing similarity measures available in the literature, as it more effectively captures the structural features of the traces.
- *Validation of structural variety measures in the context of process standardization*: The introduced measurement framework emphasizes the relationship between the reduction of structural process

variety, due to standardization efforts, and the increase in alignment costs, due to a decrease in process flexibility. The study shows that this relationship holds throughout different real-life business processes, and that structural process variety can provide valuable guidance for standardization efforts.

- *Validation of structural variety measures in the context of user behavior analysis*: The conducted eye-tracking study shows that the structural variety of users' visual scan paths serves as a significant indicator for distinguishing behavioral patterns. Furthermore, the results of the concept drift analysis highlight the effectiveness of the similarity measures introduced in Chapter 4 in detecting changes in user behavior, specifically, transitions from information search to information inference. These findings contribute novel insights into user behavior analysis and support the development of more adaptive and context-aware user assistance systems.

7.2 Limitations

This work focuses primarily on measuring structural process variety through pairwise comparisons of process traces, following the approach introduced by Weitzman [Wei92]. While this method effectively captures structural differences between traces, it does not account for the relative frequency of trace occurrences. As a result, potentially valuable information, such as whether a trace represents a common behavior or an anomaly, is not incorporated into the variety assessment. Future work could address this limitation by integrating frequency-based weighting into the clustering process, as proposed by Ackerman et al. [ABDBL12].

Another limitation of the pairwise comparison approach is its inability to evaluate structural variety at the level of individual traces. This restricts its applicability in scenarios where intra-trace behavioral changes are of interest, such as analyzing user behavior during a single comprehension task. Alternative approaches, such as the feature-based method proposed by Nehring and Puppe [NP02], offer a potential solution but require the definition of appropriate weights for each relevant structural feature.

Additionally, this work considers only the control-flow perspective of process structure. Other dimensions, such as resource and data perspectives, are not included, despite their potential to enrich the analysis of process variety [Man18]. While this focus aligns with the central role of control flow in process analysis [vdA16], future research could extend the current framework by incorporating multiple case notions into event logs [GPBvdA21], thereby enabling multi-perspective structural variety analysis.

7.3 Outlook

As shown in Chapter 3.1 the concept of structural process variety can be applied universally in BPM and process mining to generate meaningful process insights, since it relates to well-established concepts in process analysis, such as flexibility, complexity, and exploration. Furthermore, structural process variety can potentially be applied to provide valuable new insights in different application areas. In the following, three such specific application areas are described.

7.3.1 Towards a Theory on Software Customization

Software customization is a prevalent strategy in software engineering, aimed at adapting standardized software solutions to meet the specific

requirements of individual customers [MPS13, JZWL16, ASAGZ19]. This approach is particularly common in the domain of enterprise information systems, such as enterprise resource planning (ERP), and supply chain management (SCM) systems [SZG+08, MPS13, LHBF14].

Customization in software engineering is characterized by two inherently conflicting objectives:

- Standardization: The development of a uniform software product that targets a broad customer base. This strategy promotes efficient development processes, reduces maintenance overhead, and enhances security through uniformity and predictability [MPS13, JZWL16]. It also enables vendors to leverage economies of scale [SZG+08].
- Individualization: The ability to modify the standardized software to accommodate specific customer requirements, including unique workflows, specialized functionalities, and compliance with regulatory frameworks [SZG+08, JZWL16].

While individualization can enhance the utility of software for individual users, it also introduces increased complexity, higher maintenance demands, and potential security vulnerabilities. This trade-off between standardization and individualization parallels the tension between business process standardization and business process flexibility discussed in Chapter 5.

Software functionalities and requirements are commonly described at a workflow-level, reflecting the operational processes they are intended to support [SZG+08, MPS13, ASAGZ19]. The relationship between a system's customization capabilities and the variability of its supported workflows has been extensively recognized in the literature [SZG+08, MMLP09, KJ11]. In this context, structural process variety can serve as an indicator for both the degree of standardization and the adaptability of a software system. Moreover, it may also provide an estimate of

the customization effort required, including associated costs and implementation complexity.

Hence, leveraging structural process variety could provide valuable insights into key design questions, such as:

- What is the optimal degree of standardization?
- How can security be maximized without compromising adaptability?
- Which features and workflows are most critical for maintaining adaptability?

An additional dimension of software customization is software diversity [LHBF14, JZWL16], which seeks to enhance security through probabilistic defense mechanisms. Software diversity increases the difficulty of large-scale attacks by randomizing implementation details, thereby requiring attackers to target systems individually [LHBF14]. This significantly raises the cost and complexity of exploitation.

Various techniques exist for implementing software diversity [LHBF14, JZWL16], and structural process variety may offer a novel means of evaluating diversity at the control-flow level. This perspective could contribute to a more nuanced understanding of how diversity impacts both security and adaptability in software systems.

7.3.2 Governing Agentic AI Systems

Agentic artificial intelligence (AI) systems are designed to autonomously perform complex tasks with minimal human supervision [Sha23]. Recent advancements, particularly the integration of large language models (LLMs), have significantly enhanced the capabilities of such systems, enabling them to address intricate tasks across domains such as data analysis, software engineering, scientific discovery, and travel planning [Sha23, LXM+24, LHW+24, BBK+24].

As these systems become more autonomous and powerful, governance mechanisms are essential to ensure their behavior remains safe, transparent, and aligned with human intentions [Sha23]. One promising approach to maintaining control over agentic AI systems involves the specification of workflows that encode task requirements and operational constraints [LXM+24, LHW+24]. Complementing this, Besta et al. [BBK+24] introduced the concept of a *graph of thoughts*, which captures the intermediate reasoning steps of an AI agent during task execution. This approach enhances transparency by making the agent's decision-making process more comprehensible.

In this context, the analysis of structural process variety, whether in predefined workflows or observed reasoning paths, offers a valuable framework for evaluating both the restrictiveness of task constraints, as well as the degree of exploration exhibited by the agent during problem-solving. High structural variety may indicate greater flexibility and creativity, while low variety may reflect tighter control and predictability.

A related line of research has explored the application of process mining to multi-agent systems, where agent behavior is typically more constrained than in LLM-based systems [BLM22, TPK21]. Despite these constraints, the interactions among multiple agents can give rise to emergent process complexity.

Here again, structural process variety serves as a useful analytical tool. It can help delineate the scope of the action space available to agents and monitor the extent of behavioral divergence, thereby supporting both control and interpretability in complex, distributed systems.

7.3.3 Resilience of Information Systems

Resilience refers to the capacity of a system to recover from disruptions and restore its functionality to an acceptable level. In the context of enterprise information systems, resilience is of particular importance due to their central role in ensuring the timely and reliable flow of information within organizations [MKA13, WGI10]. Beyond supporting operational continuity, system resilience can also contribute to the long-term sustainability of enterprise operations by enhancing robustness and adaptability in the face of uncertainty [FvHS+22].

Resilience analysis typically considers two key dimensions [WGI10, KRvdA24]:

- the impact of a disruption on system performance,
- and the recovery duration required to return to a predefined level of functionality.

Recent research has identified process mining as a promising approach for evaluating the resilience of information systems [KRvdA24]. However, existing studies have predominantly focused on the temporal dimension of resilience, such as recovery time, while largely neglecting structural aspects.

As demonstrated in Chapter 5, structural process variety offers a valuable lens for assessing workflow adaptability. In contrast to process standardization, where structural configurations are deliberately designed and incrementally adjusted, resilience scenarios often involve unexpected disruptions that simultaneously affect multiple activities and their interdependencies. In such cases, structural process variety can serve as an indicator of a system's capacity to adapt and reconfigure its remaining functional components to maintain or restore operations.

Moreover, structural process variety can inform proactive workflow design, guiding the development of processes that are inherently more resilient, i.e., capable of maintaining functionality and adjusting dynamically under adverse conditions.

The relationship between systemic variety and resilience has been extensively studied in other disciplines, including ecology [Odu53, PH00] and organizational theory [Pag10], where diversity is recognized as a key factor in enhancing system robustness and adaptive capacity. Drawing on these insights, structural process variety may offer a novel and quantifiable means of evaluating and improving resilience in information systems.

Bibliography

[AAGK20] Cemal E. Arbatlı, Quamrul H. Ashraf, Oded Galor, and Marc Klemp. Diversity and conflict. *Econometrica*, 88(2):727–797, 2020.

[AALW23] Amine Abbad-Andaloussi, Daniel Lübke, and Barbara Weber. Conducting eye-tracking studies on large and interactive process models using eyemind. *SoftwareX*, 24:101564, 2023.

[AASS+20] Amine Abbad-Andaloussi, Pnina Soffer, Tijs Slaats, Andrea Burattin, and Barbara Weber. The impact of modularization on the understandability of declarative process models: a research model. In *Information Systems and Neuroscience: NeuroIS Retreat 2020*, pages 133–144. Springer, 2020.

[AASW24] Amine Abbad-Andaloussi, Clemens Schreiber, and Barbara Weber. On the task-specific effects of fragmentation in modular process models. In *International Conference on Conceptual Modeling*, pages 18–37. Springer, 2024.

[AAZB+21] Amine Abbad-Andaloussi, Francesca Zerbato, Andrea Burattin, Tijs Slaats, Thomas T. Hildebrandt, and Barbara Weber. Exploring how users engage with hybrid process artifacts based on declarative process models:

a behavioral analysis based on eye-tracking and think-aloud. *Software and Systems Modeling*, 20:1437–1464, 2021.

[ABDBL12] Margareta Ackerman, Shai Ben-David, Simina Brânzei, and David Loker. Weighted clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 858–863, 2012.

[ABR16] Patrick Afflerbach, Manuel Bolsinger, and Maximilian Röglinger. An economic decision model for determining the appropriate level of business process standardization. *Business Research*, 9(2):335–375, 2016.

[AC17] Samaneh Aminikhanghahi and Diane J. Cook. A survey of methods for time series change point detection. *Knowledge and information systems*, 51(2):339–367, 2017.

[Aki24] Kimberly Akinwale. Analyzing concept drift in behavioral processes. Master Thesis, Karlsruhe Institute of Technology, 2024.

[AKS+21] Sascha Alpers, Thomas Karle, Clemens Schreiber, Frank Schönthaler, and Andreas Oberweis. Process mining bei hybriden vorgehensmodellen zur umsetzung von unternehmenssoftware: Beitrag für das informatik spektrum, 2021, sonderheft "innovation in der software entwicklung". *Informatik Spektrum*, 44(3):178–189, 2021.

[All70] Frances E. Allen. Control flow analysis. *ACM Sigplan Notices*, 5(7):1–19, 1970.

[AMVW22] Adriano Augusto, Jan Mendling, Maxim Vidgof, and Bastian Wurm. The connection between process com-

plexity of event sequences and models discovered by process mining. *Information Sciences*, 598:196–215, 2022.

[AS22] Supriya Agrahari and Anil Kumar Singh. Concept drift detection in data stream mining: A literature review. *Journal of King Saud University-Computer and Information Sciences*, 34(10):9523–9540, 2022.

[ASAGZ19] Abdulrazzaq Qasem Ali, Abu Bakar Md Sultan, Abdul Azim Abd Ghani, and Hazura Zulzalil. A systematic mapping study on the customization solutions of software as a service applications. *IEEE Access*, 7:88196–88217, 2019.

[AvZRvdA23] Jan Niklas Adams, Sebastiaan J. van Zelst, Thomas Rose, and Wil M. P. van der Aalst. Explainable concept drift in process mining. *Information Systems*, 114:102177, 2023.

[Bau06] Stefan Baumgärtner. Measuring the diversity of what? and for what purpose? a conceptual comparison of ecological and economic biodiversity indices. *Available at SSRN: https://ssrn.com/abstract=894782*, 2006.

[BBK+24] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.

[BCC19] Mathilde Boltenhagen, Thomas Chatain, and Josep Carmona. Generalized alignment-based trace clustering of process behavior. In *International Conference on*

Applications and Theory of Petri Nets and Concurrency, pages 237–257. Springer, 2019.

[BDS19] Christoffer Olling Back, Søren Debois, and Tijs Slaats. Entropy as a measure of log variability. *Journal on Data Semantics*, 8:129–156, 2019.

[BEM96] Lionel Briand, Khaled El Emam, and Sandro Morasca. On the application of measurement theory in software engineering. *Empirical Software Engineering*, 1:61–88, 1996.

[BGJH09] Daniel Beimborn, Fabian Gleisner, Nils Joachim, and Andreas Hackethal. The role of process standardization in achieving it business value. In *2009 42nd Hawaii International Conference on System Sciences*, pages 1–10. IEEE, 2009.

[BHS13] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

[BL12] Michael Becker and Ralf Laue. A comparative survey of business process similarity measures. *Computers in Industry*, 63(2):148–167, 2012.

[BLM22] Rob H. Bemthuis and Sanja Lazarova-Molnar. Discovering agent models using process mining: Initial approach and a case study. In *2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/-SocialCom/SustainCom)*, pages 163–172. IEEE, 2022.

[BLS22] Andrea Burattin, Hugo A. López, and Lasse Starklit. Uncovering change: A streaming approach for declar-

ative processes. In *International Conference on Process Mining*, pages 158–170. Springer, 2022.

[BMPR12] Antonio Bucchiarone, Annapaola Marconi, Marco Pistore, and Heorhi Raik. Dynamic adaptation of fragment-based and context-aware business processes. In *2012 IEEE 19th International Conference on Web Services*, pages 33–41. IEEE, 2012.

[BMvdA13] R. P. Jagadeesh Chandra Bose, Ronny Mans, and Wil M. P. van der Aalst. Wanna improve process mining results? In *2013 IEEE symposium on computational intelligence and data mining (CIDM)*, pages 127–134. IEEE, 2013.

[BP04] Glenn J. Browne and Mitzi G. Pitts. Stopping rule use during information search in design problems. *Organizational Behavior and Human Decision Processes*, 95(2):208–224, 2004.

[BPW07] Glenn J. Browne, Mitzi G. Pitts, and James C. Wetherbe. Cognitive stopping rules for terminating information search in online tasks. *MIS quarterly*, pages 89–104, 2007.

[BPX03] Walter Bossert, Prasanta K. Pattanaik, and Yongsheng Xu. Similarity of options and the measurement of diversity. *Journal of Theoretical Politics*, 15(4):405–421, 2003.

[BRVU02] Jörg Becker, Michael Rosemann, and Christoph Von Uthmann. Guidelines of business process modeling. In *Business Process Management: Models, Techniques, and Empirical Studies*, pages 30–49. Springer, 2002.

[BS23] Christoffer O. Back and Jakob G. Simonsen. Comparing trace similarity metrics across logs and evaluation measures. In *International Conference on Advanced Information Systems Engineering*, pages 226–242. Springer, 2023.

[BSP19] Palash Bera, Pnina Soffer, and Jeffrey Parsons. Using eye tracking to expose cognitive processes in understanding conceptual models. *MIS Quarterly*, 43(4):1105–1126, 2019.

[BT03] Mary J. Benner and Michael L. Tushman. Exploitation, exploration, and process management: The productivity dilemma revisited. *Academy of management review*, 28(2):238–256, 2003.

[Bui14a] Joos Buijs. *Flexible evolutionary algorithms for mining structured process models. PhD Thesis*. Technische Universiteit Eindhoven, 2014.

[Bui14b] Joos Buijs. Receipt phase of an environmental permit application process (wabo), coselog project, 2014.

[Bur16] Andrea Burattin. Plg2: Multiperspective process randomization with online and offline simulations. In *BPM (Demos)*, pages 1–6. Citeseer, 2016.

[Bur22] Andrea Burattin. Streaming process mining. *Process Mining Handbook*, 349, 2022.

[BvdAŽP11] R. P. Jagadeesh Chandra Bose, Wil M. P. van der Aalst, Indrė Žliobaitė, and Mykola Pechenizkiy. Handling concept drift in process mining. In *Advanced Information Systems Engineering: 23rd International Conference, CAiSE 2011, London, UK, June 20-24, 2011. Proceedings 23*, pages 391–405. Springer, 2011.

[BvZS23] Alessandro Berti, Sebastiaan van Zelst, and Daniel Schuster. Pm4py: a process mining library for python. *Software Impacts*, 17:100556, 2023.

[BWBBG14] Marian Benner-Wickner, Matthias Book, Tobias Brückmann, and Volker Gruhn. Examining case management demand using event log complexity metrics. In *2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations*, pages 108–115. IEEE, 2014.

[CBB13] Sylvain Castagnos, Armelle Brun, and Anne Boyer. When diversity is needed... but not expected! In *International Conference on Advances in Information Mining and Management*, pages 44–50. IARIA XPS Press, 2013.

[CD17] Antoine Cully and Yiannis Demiris. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation*, 22(2):245–259, 2017.

[CLRS22] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2022.

[CMNR06] Jorge Cardoso, Jan Mendling, Gustaf Neumann, and Hajo A. Reijers. A discourse on complexity of process models: (survey paper). In *Business Process Management Workshops: BPM 2006 International Workshops, BPD, BPI, ENEI, GPWW, DPM, semantics4ws, Vienna, Austria, September 4-7, 2006. Proceedings 4*, pages 117–128. Springer, 2006.

[Con99] William J. Conover. *Practical nonparametric statistics*. john wiley & sons, 1999.

[CvDW22] Josep Carmona, Boudewijn van Dongen, and Matthias Weidlich. Conformance checking: foundations, milestones and challenges. In *Process mining handbook*, pages 155–190. Springer, 2022.

[CWL+19] Bogeum Choi, Austin Ward, Yuan Li, Jaime Arguello, and Robert Capra. The effects of task complexity on the use of different types of information in a search assistance tool. *ACM Transactions on Information Systems (TOIS)*, 38(1):1–28, 2019.

[CZW+16] Fang Chen, Jianlong Zhou, Yang Wang, Kun Yu, Syed Z Arshad, Ahmad Khawaji, and Dan Conway. *Robust multimodal cognitive load measurement*. Springer, 2016.

[DBDB18] Aisling J. Daly, Jan M. Baetens, and Bernard De Baets. Ecological diversity: measuring the unmeasurable. *Mathematics*, 6(7):119, 2018.

[DDG+15] Pavlos Delias, Michael Doumpos, Evangelos Grigoroudis, Panagiotis Manolitzas, and Nikolaos Matsatsinis. Supporting healthcare management decisions via robust clustering of event logs. *Knowledge-Based Systems*, 84:203–213, 2015.

[DDGB09] Remco Dijkman, Marlon Dumas, and Luciano García-Bañuelos. Graph matching algorithms for business process model similarity search. In *Business Process Management: 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings 7*, pages 48–63. Springer, 2009.

[DDVD+11] Remco Dijkman, Marlon Dumas, Boudewijn Van Dongen, Reina Käärik, and Jan Mendling. Similarity of

business process models: Metrics and evaluation. *Information Systems*, 36(2):498–516, 2011.

[DF17] Sylvain Delattre and Nicolas Fournier. On the kozachenko–leonenko entropy estimator. *Journal of Statistical Planning and Inference*, 185:69–93, 2017.

[DG01] Cheryl Dunn and Severin Grabski. An investigation of localization as an element of cognitive fit in accounting model representations. *Decision Sciences*, 32(1):55–94, 2001.

[DGP16] Claudia Diamantini, Laura Genga, and Domenico Potena. Behavioral process mining for unstructured processes. *Journal of Intelligent Information Systems*, 47:5–32, 2016.

[DJR+14] Susan Dumais, Robin Jeffries, Daniel M. Russell, Diane Tang, and Jaime Teevan. Understanding user behavior through log data and analysis. *Ways of Knowing in HCI*, pages 349–372, 2014.

[DKG+20] Andrew T. Duchowski, Krzysztof Krejtz, Nina A. Gehrer, Tanya Bafna, and Per Bækgaard. The low/high index of pupillary activity. In *CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[DM11] Matthias Dehmer and Abbe Mowshowitz. A history of graph entropy measures. *Information Sciences*, 181(1):57–78, 2011.

[dMWvdA07] Ana Karla A. de Medeiros, Anton J. M. M. Weijters, and Wil M. P. van der Aalst. Genetic process mining: an experimental evaluation. *Data mining and knowledge discovery*, 14:245–304, 2007.

[DRMR13] Marlon Dumas, L Marcello Rosa, Jan Mendling, and Hajo A. Reijers. *Fundamentals of business process management*. Springer, 2013.

[dSNF+24] Rafael Gaspar de Sousa, Antonio Carlos Meira Neto, Marcelo Fantinato, Sarajane Marques Peres, and Hajo A. Reijers. Integrated detection and localization of concept drifts in process mining with batch and stream trace clustering support. *Data & Knowledge Engineering*, 149:102253, 2024.

[DvH09] Keith Dowding and Martin van Hees. Freedom of choice. In *Handbook of Rational and Social Choice*, pages 374–292. Oxford University Press, 2009.

[DWVBVB13] Jochen De Weerdt, Seppe Vanden Broucke, Jan Vanthienen, and Bart Baesens. Active trace clustering for improved process discovery. *IEEE Transactions on Knowledge and Data Engineering*, 25(12):2708–2720, 2013.

[ESDS16] Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Information sciences*, 346:180–197, 2016.

[FAAS+24] Marco Franceschetti, Amine Abbad-Andaloussi, Clemens Schreiber, Hugo A. López, and Barbara Weber. Exploring the cognitive effects of ambiguity in process models. In *International Conference on Business Process Management*, pages 493–510. Springer, 2024.

[Fah22] Dirk Fahland. Process mining over multiple behavioral dimensions with event knowledge graphs. In *Process Mining Handbook*, pages 274–319. Springer, 2022.

[FAM18] Christian Fleig, Dominik Augenstein, and Alexander Maedche. Designing a process mining-enabled decision support system for business process standardization in erp implementation projects. In *Business Process Management Forum: BPM Forum 2018, Sydney, NSW, Australia, September 9-14, 2018, Proceedings 16*, pages 228–244. Springer, 2018.

[FG08] Alberto Fernández and Sergio Gómez. Solving non-uniqueness in agglomerative hierarchical clustering using multidendrograms. *Journal of Classification*, 25(1):43–65, 2008.

[FH07] Daniel M. Fleder and Kartik Hosanagar. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 192–199, 2007.

[Fig17] Kathrin Figl. Comprehension of procedural visual business process models: a literature review. *Business & Information Systems Engineering*, 59:41–67, 2017.

[FKLM99] Frances X. Frei, Ravi Kalakota, Andrew J. Leone, and Leslie M. Marx. Process variation as a determinant of bank performance: evidence from the retail banking study. *Management Science*, 45(9):1210–1220, 1999.

[FL15] Kathrin Figl and Ralf Laue. Influence factors for local comprehensibility of process models. *International Journal of Human-Computer Studies*, 82:96–110, 2015.

[FMR12] Catherine O. Fritz, Peter E. Morris, and Jennifer J. Richler. Effect size estimates: current use, calculations, and interpretation. *Journal of experimental psychology: General*, 141(1):2, 2012.

[FR21] Peter Fettke and Wolfgang Reisig. Modelling service-oriented systems and cloud services with heraklit. In *Advances in Service-Oriented and Cloud Computing: International Workshops of ESOCC 2020, Heraklion, Crete, Greece, September 28–30, 2020, Revised Selected Papers 8*, pages 77–89. Springer, 2021.

[FvHS+22] Andreas Fritsch, Johanna von Hammerstein, Clemens Schreiber, Stefanie Betz, and Andreas Oberweis. Pathways to greener pastures: Research opportunities to integrate life cycle assessment and sustainable business process management based on a systematic tertiary literature review. *Sustainability*, 14(18):11164, 2022.

[FVLK+17] Philippe Fournier-Viger, Jerry Chun-Wei Lin, Rage U. Kiran, Yun S. Koh, and Rincy Thomas. A survey of sequential pattern mining. *Data Science and Pattern Recognition*, 1(1):54–77, 2017.

[FZMF07] Diogo Ferreira, Marielba Zacarias, Miguel Malheiros, and Pedro Ferreira. Approaching process mining with sequence clustering: Experiments and findings. In *International conference on business process management*, pages 360–374. Springer, 2007.

[Gau11] Walter Gautschi. *Numerical analysis*. Springer Science & Business Media, 2011.

[GG12] Deepti Gupta and Shabina Ghafir. An overview of methods maintaining diversity in genetic algorithms. *International journal of emerging technology and advanced engineering*, 2(5):56–60, 2012.

[GGMH24] Steven Groß, Thomas Grisold, Jan Mendling, and Jennifer Haase. Idea generation in exploitative and ex-

plorative business process redesign techniques. *Information Systems and e-Business Management*, pages 1–29, 2024.

[GGP05] Gianluigi Greco, Antonella Guzzo, and Luigi Pontieri. Mining hierarchies of models: From abstract views to concrete specifications. In *International conference on Business Process management*, pages 32–47. Springer, 2005.

[GGPS06] Gianluigi Greco, Antonella Guzzo, Luigi Pontieri, and Domenico Sacca. Discovering expressive process models by clustering log traces. *IEEE Transactions on knowledge and data engineering*, 18(8):1010–1027, 2006.

[GH08] Andreas Glöckner and Ann-Katrin Herbold. Information processing in decisions under risk: Evidence for compensatory strategies based on automatic processes. *MPI collective goods preprint*, 2008(42), 2008.

[GH11] Andreas Glöckner and Ann-Katrin Herbold. An eye-tracking study on information processing in risky decisions: Evidence for compensatory strategies based on automatic processes. *Journal of Behavioral Decision Making*, 24(1):71–98, 2011.

[GL06] Volker Gruhn and Ralf Laue. Complexity metrics for business process models. *Business Information Systems - 9th International Conference on Business Information Systems*, pages 1–12, 2006.

[GPBvdA21] Anahita Farhang Ghahfarokhi, Gyunam Park, Alessandro Berti, and Wil M. P. van der Aalst. Ocel: a standard for object-centric event logs. In *European Confer-*

ence on Advances in Databases and Information Systems, pages 169–175. Springer, 2021.

[GR90] John Mordechai Gottman and Anup Kumar Roy. *Sequential analysis: A guide for behavioral researchers*. Cambridge University Press, 1990.

[Gra08] Nicolas Gravel. What is diversity? *Economics, rational choice and normative philosophy*, page 15, 2008.

[Gün09] Christian W. Günther. *Process mining in flexible environments*. PhD thesis, Technische Universiteit Eindhoven, 2009.

[GXTL10] Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Analysis and applications*, 13:113–129, 2010.

[GŽB+14] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

[HAC13] Amin Haji-Abolhassani and James J. Clark. A computational model for task inference in visual search. *Journal of vision*, 13(3):29–29, 2013.

[HB02] Thomas Hofmann and Joachim M. Buhmann. Pairwise data clustering by deterministic annealing. *Ieee transactions on pattern analysis and machine intelligence*, 19(1):1–14, 2002.

[HBZ+16] Cornelia Haisjackl, Irene Barba, Stefan Zugal, Pnina Soffer, Irit Hadar, Manfred Reichert, Jakob Pinggera, and Barbara Weber. Understanding declare models: strategies, pitfalls, empirical results. *Software & Systems Modeling*, 15(2):325–352, 2016.

[Hil73] Mark O. Hill. Diversity and evenness: a unifying notation and its consequences. *Ecology*, 54(2):427–432, 1973.

[HJPvdA24] Tsung-Hao Huang, Tarek Junied, Marco Pegoraro, and Wil M. P. van der Aalst. Proreco: A process discovery recommender system. In *International Conference on Advanced Information Systems Engineering*, pages 93–101. Springer, 2024.

[HK07] David A. Harrison and Katherine J Klein. What's the difference? diversity constructs as separation, variety, or disparity in organizations. *Academy of management review*, 32(4):1199–1228, 2007.

[HKP12] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2012.

[HNA+11] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. van de Weijer. *Eye Tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011.

[HW16] Marcin Hewelt and Mathias Weske. A hybrid approach for flexible case modeling and execution. In *Business Process Management Forum: BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings 14*, pages 38–54. Springer, 2016.

[IALS11] Jose Antonio Iglesias, Plamen Angelov, Agapito Ledezma, and Araceli Sanchis. Creating evolving user behavior profiles automatically. *IEEE transactions on knowledge and data engineering*, 24(5):854–867, 2011.

[IGRR09] Marta Indulska, Peter Green, Jan Recker, and Michael Rosemann. Business process modeling: Perceived

benefits. In *Conceptual Modeling-ER 2009: 28th International Conference on Conceptual Modeling, Gramado, Brazil, November 9-12, 2009. Proceedings 28*, pages 458–471. Springer, 2009.

[JCBvdA09a] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Abstractions in process mining: A taxonomy of patterns. In *International Conference on Business Process Management*, pages 159–175. Springer, 2009.

[JCBvdA09b] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Abstractions in process mining: A taxonomy of patterns. In *International Conference on Business Process Management*, pages 159–175. Springer, 2009.

[JCBvdA09c] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Context aware trace clustering: Towards improving process mining results. In *proceedings of the 2009 SIAM International Conference on Data Mining*, pages 401–412. SIAM, 2009.

[JCBvdA10] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Trace alignment in process mining: opportunities for process diagnostics. In *International Conference on Business Process Management*, pages 227–242. Springer, 2010.

[Jen05] Erica Jen. *Robust design: a repertoire of biological, ecological, and engineering case studies*. Oxford University Press, 2005.

[Joh67] Stephen C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[JVD00] Jayanth Jayaram, Shawnee K. Vickery, and Cornelia Droge. The effects of information system infrastructure

and process improvements on supply-chain time performance. *International Journal of Physical Distribution & Logistics Management*, 30(3/4):314–330, 2000.

[JZWL16] Yufei Jiang, Can Zhang, Dinghao Wu, and Peng Liu. Feature-based software customization: Preliminary analysis, formalization, and methods. In *2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE)*, pages 122–131. IEEE, 2016.

[KB19] Mahmut Kaya and Hasan Şakir Bilge. Deep metric learning: A survey. *Symmetry*, 11(9), 2019.

[KHH00] Jinwoo Kim, Jungpil Hahn, and Hyoungmee Hahn. How do we understand a system with (so) many diagrams? cognitive integration processes in diagrammatic reasoning. *Information Systems Research*, 11(3):284–303, 2000.

[KHO11] Agnes Koschmider, Thomas Hornung, and Andreas Oberweis. Recommendation-based editor for business process modeling. *Data & Knowledge Engineering*, 70(6):483–503, 2011.

[Kin23] Nina Kinkelin. Untersuchung zur anwendbarkeit von prozesssimulation im rahmen von process mining. Bachelor Thesis, Karlsruhe Institute of Technology, 2023.

[KJ11] Jaap Kabbedijk and Slinger Jansen. Variability in multi-tenant environments: Architectural design patterns from industry. In *Advances in Conceptual Modeling. Recent Developments and New Directions: ER 2011 Workshops FP-UML, MoRE-BI, Onto-CoM, SeCoGIS, Variabil-*

ity@ ER, WISM, Brussels, Belgium, October 31-November 3, 2011. Proceedings 30, pages 151–160. Springer, 2011.

[KNGW21] Martin Kabierski, Hoang Lam Nguyen, Lars Grunske, and Matthias Weidlich. Sampling what matters: relevance-guided sampling of event logs. In *2021 3rd International Conference on Process Mining (ICPM)*, pages 64–71. IEEE, 2021.

[Kos19] Sven Kosub. A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters*, 120:36–38, 2019.

[KP17] Matevž Kunaver and Tomaž Požrl. Diversity in recommender systems–a survey. *Knowledge-based systems*, 123:154–162, 2017.

[Kri88] Jürgen Kriz. *Facts and artefacts in social science: An epistemological and methodological analysis of empirical social science research techniques*. McGraw-Hill, 1988.

[KRvdA24] Alexander Kraus, Jana-Rebecca Rehse, and Han van der Aa. Data-driven assessment of business process resilience. *Process Science*, 1(1):4, 2024.

[KSDK14] Krzysztof Krejtz, Tomasz Szmidt, Andrew T. Duchowski, and Izabela Krejtz. Entropy-based statistical analysis of eye movement transitions. In *Proceedings of the symposium on eye tracking research and applications*, pages 159–166, 2014.

[KSP05] Alla Keselman, Laura Slaughter, and Vimla L. Patel. Toward a framework for understanding lay public's comprehension of disaster and bioterrorism information. *Journal of Biomedical Informatics*, 38(4):331–344, 2005.

[KSR23] Nina Kinkelin, Clemens Schreiber, and Josua Reimold. Combining process mining and simulation to guide users towards process improvement in mpmx. In *Proc. of CEUR Workshop on ICPM Demo Track-FORTHCOMING*, 2023.

[KWW11] Matthias Kunze, Matthias Weidlich, and Mathias Weske. Behavioral similarity–a proper metric. In *Business Process Management: 9th International Conference, BPM 2011, Clermont-Ferrand, France, August 30-September 2, 2011. Proceedings 9*, pages 166–181. Springer, 2011.

[KWWL14] Julian Krumeich, Benjamin Weis, Dirk Werth, and Peter Loos. Event-driven business process management: where are we now?: A comprehensive synthesis and analysis of literature. *Business Process Management Journal*, 20(4):615–633, 2014.

[LCB14] Amaury L'Huillier, Sylvain Castagnos, and Anne Boyer. Understanding usages by modeling diversity over time. In *22nd Conference on user modeling, adaptation, and personalization*, volume 1181, 2014.

[LCX+13] Ying Li, Bin Cao, Lida Xu, Jianwei Yin, Shuiguang Deng, Yuyu Yin, and Zhaohui Wu. An efficient recommendation method for improving business process modeling. *IEEE Transactions on Industrial Informatics*, 10(1):502–513, 2013.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.

[LHBF14] Per Larsen, Andrei Homescu, Stefan Brunthaler, and Michael Franz. Sok: Automated software diversity. In *2014 IEEE Symposium on Security and Privacy*, pages 276–291. IEEE, 2014.

[LHW+24] Zelong Li, Wenyue Hua, Hao Wang, He Zhu, and Yongfeng Zhang. Formal-llm: Integrating formal language and natural language for controllable llm-based agents. *arXiv preprint arXiv:2402.00798*, 2024.

[LLD+18] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018.

[LM05] Antonella Longo and Gianmario Motta. Design processes for sustainable performances: a model and a method. In *International Conference on Business Process Management*, pages 399–407. Springer, 2005.

[LRWM+11] Marcello La Rosa, Petia Wohed, Jan Mendling, Arthur H. M. Ter Hofstede, Hajo A. Reijers, and Wil M. P. van der Aalst. Managing process model complexity via abstract syntax modifications. *IEEE Transactions on Industrial Informatics*, 7(4):614–629, 2011.

[LS87] Jill H. Larkin and Herbert A. Simon. Why a diagram is (sometimes) worth ten thousand words. *Cognitive science*, 11(1):65–100, 1987.

[LS11] Joel Lehman and Kenneth O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.

[LvdA14] Maikel Leemans and Wil M. P. van der Aalst. Discovery of frequent episodes in event logs. In *International*

symposium on data-driven process discovery and analysis, pages 1–31. Springer, 2014.

[LvdABP21] Sander J. J. Leemans, Wil M. P. van der Aalst, Tobias Brockhoff, and Artem Polyvyanyy. Stochastic process mining: Earth movers' stochastic conformance. *IS*, 102:101724, 2021.

[LWB19] Loet Leydesdorff, Caroline S. Wagner, and Lutz Bornmann. Interdisciplinarity as diversity in citation patterns among journals: Rao-stirling diversity, relative variety, and the gini coefficient. *Journal of informetrics*, 13(1):255–269, 2019.

[LXM+24] Zelong Li, Shuyuan Xu, Kai Mei, Wenyue Hua, Balaji Rama, Om Raheja, Hao Wang, He Zhu, and Yongfeng Zhang. Autoflow: Automated workflow generation for large language model agents. *arXiv preprint arXiv:2407.12821*, 2024.

[LZ76] Abraham Lempel and Jacob Ziv. On the complexity of finite sequences. *IEEE Transactions on information theory*, 22(1):75–81, 1976.

[Mag13] Anne E. Magurran. *Ecological diversity and its measurement*. Springer Science & Business Media, 2013.

[Man18] Felix Mannhardt. Multi-perspective process mining. *Technische Universiteit Eindhoven*, 2018.

[Mar06] Gary Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

[May89] Richard E. Mayer. Human nonadversary problem solving. *Human and machine problem solving*, pages 39–56, 1989.

[McC76] Thomas J. McCabe. A complexity measure. *IEEE Transactions on software Engineering*, (4):308–320, 1976.

[McC00] Kevin S. McCann. The diversity–stability debate. *Nature*, 405(6783):228–233, 2000.

[Men08] Jan Mendling. *Metrics for process models: empirical foundations of verification, error prediction, and guidelines for correctness*, volume 6. Springer Science & Business Media, 2008.

[MEW10] Björn Münstermann, Andreas Eckhardt, and Tim Weitzel. The performance impact of business process standardization: An empirical evaluation of the recruitment process. *Business Process Management Journal*, 16(1):29–56, 2010.

[MFF+23] Oscar Manglaras, Alex Farkas, Peter Fule, Christoph Treude, and Markus Wagner. Problems in microservice development: Supporting visualisation. In *2023 IEEE Working Conference on Software Visualization (VISSOFT)*, pages 62–72. IEEE, 2023.

[MGC10] Jorge Munoz-Gama and Josep Carmona. A fresh look at precision in process conformance. In *International Conference on Business Process Management*, pages 211–226. Springer, 2010.

[MKA13] Günter Müller, Thomas G. Koslowski, and Rafael Accorsi. Resilience-a new research field in business information systems? In *Business Information Systems Workshops: BIS 2013 International Workshops, Poznań, Poland, June 19-20, 2013, Revised Papers 16*, pages 3–14. Springer, 2013.

[MM21] Monika M. Mandelburger and Jan Mendling. Cognitive diagram understanding and task performance in systems analysis and design. *MIS Quarterly*, 45(4):2101–2157, 2021.

[MMLP09] Ralph Mietzner, Andreas Metzger, Frank Leymann, and Klaus Pohl. Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications. In *2009 ICSE Workshop on Principles of Engineering Service Oriented Systems*, pages 18–25. IEEE, 2009.

[Mor02] Craig Moritz. Strategies to protect biological diversity and the evolutionary processes that sustain it. *Systematic biology*, 51(2):238–254, 2002.

[MPS13] Marko Mijac, Ruben Picek, and Zlatko Stapic. Cloud erp system customization challenges. In *Central European Conference on Information and Intelligent Systems*, page 132. Faculty of Organization and Informatics Varazdin, 2013.

[MRC07] Jan Mendling, Hajo A. Reijers, and Jorge Cardoso. What makes process models understandable? In *Business Process Management: 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007. Proceedings 5*, pages 48–63. Springer, 2007.

[MRD+15] Thomas Molka, David Redlich, Marc Drobek, Xiao-Jun Zeng, and Wasif Gilani. Diversity guided evolutionary mining of hierarchical process models. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 1247–1254, 2015.

[MRvdA10] Jan Mendling, Hajo A. Reijers, and Wil M. P. van der Aalst. Seven process modeling guidelines (7pmg). *Information and software technology*, 52(2):127–136, 2010.

[NP02] Klaus Nehring and Clemens Puppe. A theory of diversity. *Econometrica*, 70(3):1155–1198, 2002.

[NP09] Klaus Nehring and Clemens Puppe. Diversity. *The handbook of rational and social choice*, pages 298–322, 2009.

[NSA14] Gustaf Neumann, Stefan Sobernig, and Michael Aram. Evolutionary business information systems: Perspectives and challenges of an emerging class of information systems. *Business & information systems engineering*, 6:33–38, 2014.

[NW70] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[Odu53] Eugene Odum. *Fundamentals of ecology*, volume 1. Saunders Philadelphia, 1953.

[OMLR+16] Alireza Ostovar, Abderrahmane Maaradji, Marcello La Rosa, Arthur H. M. ter Hofstede, and Boudewijn F. V. van Dongen. Detecting drift from event streams of unpredictable business processes. In *Conceptual Modeling: 35th International Conference, ER 2016, Gifu, Japan, November 14-17, 2016, Proceedings 35*, pages 330–346. Springer, 2016.

[Pag10] Scott E. Page. *Diversity and complexity*. Princeton University Press, 2010.

[Par72] David L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, 1972.

[PDR+17] Jihyun Park, Kameryn Denaro, Fernando Rodriguez, Padhraic Smyth, and Mark Warschauer. Detecting changes in student behavior from clickstream data. In *Proceedings of the seventh international learning analytics & knowledge conference*, pages 21–30, 2017.

[Pen03] Brian T. Pentland. Conceptualizing and measuring variety in the execution of organizational work processes. *Management Science*, 49(7):857–870, 2003.

[PH00] Andy Purvis and Andy Hector. Getting the measure of biodiversity. *Nature*, 405(6783):212–219, 2000.

[PLKH20] Brian T. Pentland, Peng Liu, Waldemar Kremser, and Thorvald Hærem. The dynamics of drift in digitized processes. *MIS quarterly*, 44(1), 2020.

[PM13] Razvan Petrusel and Jan Mendling. Eye-tracking the factors of process model comprehension tasks. In Camille Salinesi, Moira C. Norrie, and Óscar Pastor, editors, *Advanced Information Systems Engineering*, pages 224–239, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[PMR16] Razvan Petrusel, Jan Mendling, and Hajo A. Reijers. Task-specific visual cues for improving process model understanding. *Information and Software Technology*, 79:63–78, 2016.

[PMR17] Razvan Petrusel, Jan Mendling, and Hajo A. Reijers. How visual cognition influences process model comprehension. *Decision Support Systems*, 96:1–16, 2017.

[PRM16] Johannes Pflug and Stefanie Rinderle-Ma. Process instance similarity: Potentials, metrics, applications. In *On the Move to Meaningful Internet Systems: OTM 2016 Conferences*, pages 136–154. Springer, 2016.

[PRP12] Marco Porta, Stefania Ricotti, and Calet J. Perez. Emotional e-learning through eye tracking. In *Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON)*, pages 1–6. IEEE, 2012.

[PSvdA07] Maja Pesic, Helen Schonenberg, and Wil M. P. van der Aalst. Declare: Full support for loosely-structured processes. In *11th IEEE international enterprise distributed object computing conference (EDOC 2007)*, pages 287–287. IEEE, 2007.

[PTTVG03] Fred Paas, Juhani E Tuovinen, Huib Tabbers, and Pascal WM Van Gerven. Cognitive load measurement as a means to advance cognitive load theory. *Educational psychologist*, 2003.

[PX90] Prasanta K. Pattanaik and Yongsheng Xu. On ranking opportunity sets in terms of freedom of choice. *Recherches Économiques de Louvain/Louvain Economic Review*, 56(3-4):383–390, 1990.

[Rao82] C. Radhakrishna Rao. Diversity and dissimilarity coefficients: a unified approach. *Theoretical population biology*, 21(1):24–43, 1982.

[RBT15] Narayan Ramasubbu, Anandhi Bharadwaj, and Giri K. Tayi. Software process diversity. *Mis Quarterly*, 39(4):787–808, 2015.

[RDG+15] Heidi L. Romero, Remco M. Dijkman, Paul W. P. J. Grefen, Arjan J. Van Weele, and Ad de Jong. Measures

of process harmonization. *Information and Software Technology*, 63:31–43, 2015.

[RDGvW15] Heidi L. Romero, Remco M. Dijkman, Paul W. P. J. Grefen, and Arjan J. van Weele. Factors that determine the extent of business process standardization and the subsequent effect on business performance. *Business & Information Systems Engineering*, 57:261–270, 2015.

[RHB14] Manfred Reichert, Alena Hallerbach, and Thomas Bauer. Lifecycle management of business process variants. In *Handbook on business process management 1: introduction, methods, and information systems*, pages 251–278. Springer, 2014.

[RJMR19] Hamzah Ritchi, Mieke J. Jans, Jan Mendling, and Hajo A. Reijers. The influence of business process representation on performance of different task types. *Journal of Information Systems*, 2019.

[RL15] R. Riedl and P.M. Léger. *Fundamentals of NeuroIS: Information Systems and the Brain*. Studies in Neuroscience, Psychology and Behavioral Economics. Springer Berlin Heidelberg, 2015.

[RM08] Hajo A. Reijers and Jan Mendling. Modularity in process models: Review and effects. In *BPM*, volume 8, pages 20–35. Springer, 2008.

[RMD11] Hajo A. Reijers, Jan Mendling, and Remco M. Dijkman. Human and automatic modularizations of process models to enhance their comprehension. *Information Systems*, 36(5):881–897, 2011.

[Rob79] Fred S. Robert. *Measurement theory with applications to decision making, utility, and the social sciences*. Addison-Wesley, 1979.

[RRvdW14] Jan Recker, Hajo A. Reijers, and Sander G. van de Wouw. Process model comprehension: the effects of cognitive abilities, learning style, and strategy. *Communications of the association for information systems*, 34(1):9, 2014.

[RSM+10] Christoph Rosenkranz, Stefan Seidel, Jan Mendling, Markus Schaefermeyer, and Jan Recker. Towards a framework for business process standardization. In *Business Process Management Workshops: BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers 7*, pages 53–63. Springer, 2010.

[RW12] Manfred Reichert and Barbara Weber. *Enabling flexibility in process-aware information systems: challenges, methods, technologies*, volume 54. Springer, 2012.

[SAA24] Clemens Schreiber and Amine Abbad-Andaloussi. Structural process variety and standardization. In *ICPM, forthcoming*, 2024.

[SAAW23] Clemens Schreiber, Amine Abbad-Andaloussi, and Barbara Weber. On the cognitive effects of abstraction and fragmentation in modularized process models. In *International Conference on Business Process Management*, pages 359–376. Springer, 2023.

[SAAW24] Clemens Schreiber, Amine Abbad-Andaloussi, and Barbara Weber. On the cognitive and behavioral effects of abstraction and fragmentation in modularized process models. *Information Systems*, 125:102424, 2024.

[SAS11] John Sweller, Paul Ayres, and Kalyuga Slava. *Cognitive load theory*. Springer, 2011.

[SC94] John Sweller and Paul Chandler. Why some material is difficult to learn. *Cognition and instruction*, 12(3):185–233, 1994.

[SCBS08] Frederick Shic, Katarzyna Chawarska, Jessica Bradshaw, and Brian Scassellati. Autism, eye-tracking, entropy. In *2008 7th IEEE International Conference on Development and Learning*, pages 73–78. IEEE, 2008.

[SDFBS21] Denise M. V. Sato, Sheila Cristiana De Freitas, Jean P. Barddal, and Edson E. Scalabrin. A survey on concept drift in process mining. *ACM Computing Surveys (CSUR)*, 54(9):1–38, 2021.

[Sel74] Peter H. Sellers. On the theory and computation of evolutionary distances. *SIAM Journal on Applied Mathematics*, 26(4):787–793, 1974.

[SGvdA09] Minseok Song, Christian W. Günther, and Wil M. P. van der Aalst. Trace clustering in process mining. In *Business Process Management Workshops: BPM 2008 International Workshops, Milano, Italy, September 1-4, 2008. Revised Papers 6*, pages 109–120. Springer, 2009.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[Sha23] Yonadav et al. Shavit. Practices for governing agentic ai systems. *Research Paper, OpenAI*, 2023.

[SMH+03] Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest

neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4):301–321, 2003.

[SMN+10] Graeme Shanks, Daniel Moody, Jasmina Nuredini, Daniel Tobin, and Ron Weber. Representing classes of things and properties in general in conceptual modelling: An empirical evaluation. *Journal of Database Management (JDM)*, 21(2):1–25, 2010.

[SMR+08a] Helen Schonenberg, Ronny Mans, Nick Russell, Nataliya Mulyar, and Wil M. P. van der Aalst. Process flexibility: A survey of contemporary approaches. In *International Workshop on Cooperation and Interoperability, Architecture and Ontology*, pages 16–30. Springer, 2008.

[SMR+08b] Helen Schonenberg, Ronny Mans, Nick Russell, Nataliya Mulyar, and Wil M. P. van der Aalst. Towards a taxonomy of process flexibility. In *CAiSE forum*, volume 344, pages 81–84, 2008.

[Som18] Ian Sommerville. *Software engineering*. Pearson Studium, 2018.

[Ste46] Stanley S. Stevens. On the theory of scales of measurement. *Science*, 103(2684):677–680, 1946.

[STF+17] Andreas Schoknecht, Tom Thaler, Peter Fettke, Andreas Oberweis, and Ralf Laue. Similarity of business process models-a state-of-the-art analysis. *ACM CSUR*, 50(4):1–33, 2017.

[Sti07] Andy Stirling. A general framework for analysing diversity in science, technology and society. *Journal of the Royal Society interface*, 4(15):707–719, 2007.

[SZG+08] Wei Sun, Xin Zhang, Chang Jie Guo, Pei Sun, and Hui Su. Software as a service: Configuration and customization perspectives. In *2008 ieee congress on services part ii (services-2 2008)*, pages 18–25. IEEE, 2008.

[TLRDM21] Farbod Taymouri, Marcello La Rosa, Marlon Dumas, and Fabrizio M. Maggi. Business process variant analysis: Survey and classification. *Knowledge-Based Systems*, 211:106557, 2021.

[TOJC23] Gabriel M. Tavares, Rafael S. Oyamada, Sylvio Barbon Junior, and Paolo Ceravolo. Trace encoding in process mining: A survey and benchmarking. *Engineering Applications of Artificial Intelligence*, 126:107028, 2023.

[TOV20] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

[TPK21] Andrei Tour, Artem Polyvyanyy, and Anna Kalenkova. Agent system mining: vision, benefits, and challenges. *IEEE Access*, 9:99480–99494, 2021.

[Tre14] Roger Tregear. Business process standardization. In *Handbook on business process management 2: Strategic alignment, governance, people and culture*, pages 421–441. Springer, 2014.

[TRV+16] Oktay Turetken, Tessa Rompen, Irene Vanderfeesten, Ahmet Dikici, and Jan van Moll. The effect of modularity representation and presentation medium on the understandability of business process models in bpmn. In *Business Process Management: 14th International Conference, BPM 2016, Rio de Janeiro, Brazil, September 18-22, 2016. Proceedings 14*, pages 289–307. Springer, 2016.

[TSHvdA16] Niek Tax, Natalia Sidorova, Reinder Haakma, and Wil M. P. van der Aalst. Mining local process models. *Journal of Innovation in Digital Ecosystems*, 3(2):183–196, 2016.

[TSHvdA18] Niek Tax, Natalia Sidorova, Reinder Haakma, and Wil M. P. van der Aalst. Event abstraction for process mining using supervised learning techniques. In *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 1*, pages 251–269. Springer, 2018.

[Urs02] Rasmus K. Ursem. Diversity-guided evolutionary algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 462–471. Springer, 2002.

[Vak99] Pertti Vakkari. Task complexity, problem structure and information actions: Integrating studies on information seeking and retrieval. *Information processing & management*, 35(6):819–837, 1999.

[vB24] Jan et al. vom Brocke. Process science: the interdisciplinary study of socio-technical change. *Process Science*, 1(1):1, 2024.

[VBR14] Jan Vom Brocke and Michael Rosemann. *Handbook on business process management 1: Introduction, methods, and information systems*. Springer, 2014.

[vD15] Boudewijn van Dongen. Bpi challenge 2015, https://doi.org/10.4121/uuid:31a308ef-c844-48da-948c-305d167a0ec1. *4TU.ResearchData*, 2015.

[vD17] Boudewijn van Dongen. Bpi challenge 2017, https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b. *4TU.ResearchData*, 2017.

[vD19] Boudewijn van Dongen. Bpi challenge 2019, https://doi.org/10.4121/uuid:d06aff4b-79f0-45e6-8ec8-e19730c248f1. In *4TU.ResearchData*, 2019.

[vdA12] Wil M. P. et al. van der Aalst. Process mining manifesto. In *Business Process Management Workshops: BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part I 9*, pages 169–194. Springer, 2012.

[vdA16] Wil M. P. van der Aalst. *Process mining: data science in action*, volume 2. Springer, 2016.

[vdA18] Wil M. P. van der Aalst. Process mining and simulation: A match made in heaven! In *SummerSim*, pages 4–1, 2018.

[vdA20] Wil M. P. van der Aalst. On the pareto principle in process mining, task mining, and robotic process automation. In *DATA*, pages 5–12, 2020.

[vdA22] Wil M. P. van der Aalst. Process mining: a 360 degree overview. In *Process Mining Handbook*, pages 3–34. Springer, 2022.

[vdAJ00] Wil M. P. van der Aalst and Stefan Jablonski. Dealing with workflow change: identification of issues and solutions. *Computer systems science and engineering*, 15(5):267–276, 2000.

[vdATHKB03] Wil M. P. van der Aalst, Arthur H. M. Ter Hofstede, Bartek Kiepuszewski, and Alistair P. Barros. Workflow patterns. *Distributed and parallel databases*, 14:5–51, 2003.

[vdAWG05] Wil M. P. van der Aalst, Mathias Weske, and Dolf Grünbauer. Case handling: a new paradigm for busi-

ness process support. *Data & knowledge engineering*, 53(2):129–162, 2005.

[VdB08] Jeroen C. J. M. Van den Bergh. Optimal diversity: increasing returns versus recombinant innovation. *Journal of Economic Behavior & Organization*, 68(3-4):565–580, 2008.

[vDMKR23] Christopher van Dun, Linda Moder, Wolfgang Kratsch, and Maximilian Röglinger. Processgan: Supporting the creation of business process improvement ideas through generative machine learning. *Decision Support Systems*, 165:113880, 2023.

[Ves91] Iris Vessey. Cognitive fit: A theory-based analysis of the graphs versus tables literature. *Decision sciences*, 22(2):219–240, 1991.

[VG91] Iris Vessey and Dennis Galletta. Cognitive fit: An empirical study of information acquisition. *Information systems research*, 2(1):63–84, 1991.

[VLS16] Amy Van Looy and Aygun Shafagatova. Business process performance measurement: a structured literature review of indicators, measures and metrics. *SpringerPlus*, 5(1):1797, 2016.

[VW93] Paul F. Velleman and Leland Wilkinson. Nominal, ordinal, interval, and ratio typologies are misleading. *The American Statistician*, 47(1):65–72, 1993.

[VWM23] Maxim Vidgof, Bastian Wurm, and Jan Mendling. The impact of process complexity on process performance: A study using event log data. In *International Conference on Business Process Management*, pages 413–429. Springer, 2023.

[WCI+22] Wei Wang, Tianwa Chen, Marta Indulska, Shazia Sadiq, and Barbara Weber. Business process and rule integration approaches-an empirical analysis of model understanding. *Information Systems*, 104:101901, 2022.

[Wei92] Martin L. Weitzman. On diversity. *The quarterly journal of economics*, 107(2):363–405, 1992.

[WFR21] Barbara Weber, Thomas Fischer, and René Riedl. Brain and autonomic nervous system activity measurement in software engineering: A systematic literature review. *Journal of Systems and Software*, 178, 2021.

[WGI10] J. W. Wang, F. Gao, and W. H. Ip. Measurement of resilience and its application to enterprise information systems. *Enterprise information systems*, 4(2):215–223, 2010.

[WKBW09] Kim Wüllenweber, Wolfgang Koenig, Daniel Beimborn, and Tim Weitzel. The impact of process standardization on business process outsourcing success. *Information systems outsourcing: enduring themes, global challenges, and process opportunities*, pages 527–548, 2009.

[WNP+23] Michael Winter, Heiko Neumann, Rüdiger Pryss, Thomas Probst, and Manfred Reichert. Defining gaze patterns for process model literacy–exploring visual routines in process models with diverse mappings. *Expert Systems with Applications*, 213:119217, 2023.

[Wol94] Jeremy M. Wolfe. Guided search 2.0 a revised model of visual search. *Psychonomic bulletin & review*, 1:202–238, 1994.

[WPP+22] Michael Winter, Rüdiger Pryss, Thomas Probst, Julia Baß, and Manfred Reichert. Measuring the cognitive

complexity in the comprehension of modular process models. *IEEE Transactions on Cognitive and Developmental Systems*, 14(1):164–180, 2022.

[WSMF18] Bastian Wurm, Theresa Schmiedel, Jan Mendling, and Christian Fleig. Development of a measurement scale for business process standardization. In *ECIS*, 2018.

[WWPP06] Yingxu Wang, Ying Wang, Shushma Patel, and Dilip Patel. A layered reference model of the brain (lrmb). *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 36(2):124–133, 2006.

[WZT+16] Gang Wang, Xinyi Zhang, Shiliang Tang, Haitao Zheng, and Ben Y. Zhao. Unsupervised clickstream clustering for user behavior analysis. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 225–236, 2016.

[YB07] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007.

[YER22] Mansoureh Yari Eili and Jalal Rezaeenour. A survey on recommendation in process mining. *Concurrency and Computation: Practice and Experience*, 34(26):e7304, 2022.

[zMR13] Michael zur Muehlen and Jan Recker. How much language is enough? theoretical and practical use of the business process modeling notation. *Seminal Contributions to Information Systems Engineering: 25 Years of CAiSE*, pages 429–443, 2013.

[ŽPG16] Indrė Žliobaitė, Mykola Pechenizkiy, and Joao Gama. An overview of concept drift applications. *Big data analysis: new algorithms for a new society*, pages 91–114, 2016.

[ZRSH20] Fareed Zandkarimi, Jana-Rebecca Rehse, Pouya Soudmand, and Hartmut Hoehle. A generic framework for trace clustering in process mining. In *2020 2nd International Conference on Process Mining (ICPM)*, pages 177–184. IEEE, 2020.

[ZSDF+21] Francesca Zerbato, Ronny Seiger, Gemma Di Federico, Andrea Burattin, and Barbara Weber. Granularity in process mining: Can we fix it? In *International Workshop on BPM Problems to Solve Before We Die*, pages 40–44. CEUR-WS, 2021.

[ZSW21] Francesca Zerbato, Pnina Soffer, and Barbara Weber. Initial insights into exploratory process mining practices. In *Business Process Management Forum: BPM Forum 2021, Rome, Italy, September 06–10, 2021, Proceedings 19*, pages 145–161. Springer, 2021.

[Zug13] Stefan Zugal. *Applying cognitive psychology for improving the creation, understanding and maintenance of business process models*. PhD thesis, University of Innsbruck, 2013.

[Zus91] Horst Zuse. *Software complexity: measures and methods*, volume 4. Walter de Gruyter GmbH & Co KG, 1991.

[ZWW+10] Haiping Zha, Jianmin Wang, Lijie Wen, Chaokun Wang, and Jiaguang Sun. A workflow net similarity measure based on transition adjacency relations. *Computers in Industry*, 61(5), 2010.

Publication Overview

Some of the chapters in this thesis reflect the content of previously published and submitted articles by the author Clemens Schreiber.

List of articles

Chapter 3 and Chapter 5 relate to the following publication:

- Schreiber, C., and Abbad-Andaloussi, A. (2024). Structural process variety and standardization. In 2024 6th International Conference on Process Mining (pp. 153-160). IEEE.

Chapter 4 relates to the following publication (published after the submission of the thesis):

- Schreiber, C., Abbad-Andaloussi, A., Burattin, A., Oberweis, A., and Weber, B. (2026). Graph-based similarity measures for the structural comparison of process traces. Information Systems, 102671.

Chapter 6 relates to the following publications:

- Schreiber, C., Abbad-Andaloussi, A., and Weber, B. (2023). On the cognitive effects of abstraction and fragmentation in modularized process models. In International Conference on Business Process Management (pp. 359-376). Cham: Springer Nature Switzerland.

- Schreiber, C., Abbad-Andaloussi, A., and Weber, B. (2024). On the cognitive and behavioral effects of abstraction and fragmentation in modularized process models. Information Systems, 102424.

The author served as the lead author in all publications. His contributions encompassed drafting the original manuscripts, revising and editing content, creating visualizations, conducting both empirical and formal analyses, developing the research methodology, and shaping the overall conceptual framework of the studies.

Nomenclature

Process Mining

SYMBOL	DESCRIPTION
$\mathcal{A}$	universe of activities
$\mathcal{C}$	universe of case ids
$\mathcal{E}$	universe of events
$\mathcal{F}$	universe of features
$\mathcal{G}$	universe of process graphs
$\mathcal{T}$	universe of timestamps
$\mathcal{A}^*$	universe of traces
$\mathbf{V}$	vector
A	set of activities
C	cluster of traces, i.e., a subset of a flat event log
D	set of directly follows relations between activities
D'_L	distance matrix calculated based on the pairwise comparison of all traces in a flat event log
F_e	set of edge features
F_n	set of n-grams
G	a directed graph
N	set of nodes that belong to a directed graph

E	set of edges that belong to a directed graph
L	event log
L'	flat event log
V	function to measure structural process variety
V_{Prefix}	structural process variety measure based on graph entropy and prefix automata
V_{Knn}	structural process variety measure based on K-nearest neighbor
$V_{\text{N-block}}$	structural process variety measure based on n-grams
$V_{\text{Lempel-Ziv}}$	structural process variety measure based on Lempel-Ziv complexity measure
$V_{\text{Levenshtein}}$	structural process variety measure based on Levenshtein distance
V_{Weitzman}	structural process variety measure based on Weitzman diversity measure
a	activity
a_k	k^{th} activity in a trace
c	case id
d	function to measure the distance between two traces
d_{avg}	average linkage criterion based on the average distance between two trace clusters
d_{max}	complete linkage criterion based on the maximum distance between two trace clusters
d_{min}	single linkage criterion based on the minimum distance between two trace clusters
d	function to measure the distance between two traces

d_C	function to measure the cosine distance between two traces
d_E	function to measure the Euclidean distance between two traces
d_J	function to measure the Jaccard distance between two traces
e	event
h	function to define a set of features for a trace
h_e	function to define a set of edge features for a trace
h_n	function to define a set of n-grams for a trace
s	function to measure similarity between two traces
s_{Edit}	function to measure the edit distance between two traces
t	timestamp
v	element in a vector
w	function mapping a set of trace features into a vector space
σ	trace
λ	subsequence of a trace
$\mathcal{B}$	multiset
$\mathcal{P}$	power set
$\#_{act}(e)$	mapping from event to activity
$\#_{case}(e)$	mapping from event to case id
$\#_{time}(e)$	mapping from event to timestamp
$\|\sigma\|$	length of a trace
$\|F_i \cup F_j\|$	size of the union of two sets of features
$\mathbb{R}_0^+$	set of positive real numbers including zero
$\mathbb{R}^n$	set of all vectors of real numbers including zero

Agg	summation of similarity measures
C1	cosine distance based on 1-gram
C2	cosine distance based on 2-gram
C3	cosine distance based on 3-gram
Cg	graph-based cosine similarity
CMR	cosine distance based on maximal repeat alphabet
E1	Euclidean distance based on 1-gram
E2	Euclidean distance based on 2-gram
E3	Euclidean distance based on 3-gram
EF	trace similarity measure based on eventually-follows relationships
EMR	Euclidean distance based on maximal repeat alphabet
J1	Jaccard distance based on 1-gram
J2	Jaccard distance based on 2-gram
J3	Jaccard distance based on 3-gram
JMR	Jaccard distance based on maximal repeat alphabet
Jg	graph-based Jaccard similarity
LD	Levenshtein distance
N-LD	normalized Levenshtein distance
BPM	Business Process Management

Measurement Theory

SYMBOL	DESCRIPTION

P	set of partial properties
R	relations on empirical objects
R_E	empirical relational system
R_F	formal relational system
R	relations on formal objects
O	empirical object
g	admissible transformation
p	partial property
σ'	trace after atomic modification
μ	measure
$\circ$	binary operations on empirical objects
$\bullet$	closed binary operations on formal objects
$\mathbb{R}$	set of real numbers

Diversity Analysis

SYMBOL	DESCRIPTION
F	set of features
S	set of variants
D	function to measures the diversity of a system
D_{Entropy}	diversity measure based on generalized entropy function
$D_{\text{Nehring-Puppe}}$	diversity measure based on weighted features
D_{Number}	diversity measure based on the number of variants within a set

D_{Rao}	diversity measure based on the combination of pairwise distances between variants and their relative frequencies
$D_{\text{Shannon-Wiener}}$	diversity measure based on Shannon-Wiener entropy function
D_{Weitzman}	diversity measure based on pairwise distances between variants
a_i	number of a variant within a set
f	feature
n	number of variants contained in a set that belongs to a system
m	number of features that allow to distinguish between variants in a set
s	variant
x_{ij}	description of a variant in terms of features
p_i	proportion (relative frequency) of a variant within a set
Ω	system
λ_f	weights over features
$\mathbb{N}$	set of natural numbers

Eye-tracking Analysis

SYMBOL	DESCRIPTION
$LHIPA$	low/high index of pupil activity
MAE	mean absolute error
MSD	mean signed difference
r^2	effect size

List of Figures

1.1 Graphical representation of the trace $\sigma = \langle A, B, D, A, C\ D, A, D \rangle$, consisting of the four activities A, B, C, and D. The trace contains three structural patterns indicated by the colored edges: a loop (l), a choice (c), and a skip (s). 3

2.1 Example for the application of process discovery. 12
2.2 Graphical representation of the four structural patterns: (i) sequence, (ii) loop, (iii) choice, and (iv) skip. 14
2.3 Two sets of means of transportation, differing in terms of number of variants, relative variant frequencies, and the distinctiveness of the variants. The figure is based on Purvis and Hector [PH00]. 27
2.4 An example of (i) a taxonomic tree describing the evolutionary relations between six different species. The ultrametric distances between the species are depicted by (ii) the distance matrix. In the example, originally provided by Weitzman [Wei92], the taxonomic tree represents the following species: 1 - human, 2 - common chimpanzee, 3 - gorilla, 4 - orangutan, 5 - common gibbon, and 6 - siamang gibbon. The figure is adapted from Weitzman [Wei92]. 39

3.1 Four types of concept drift: (i) *sudden drift*, (ii) *recurring drift*, (ii) *gradual drift*, and (iv) *incremental drift*. L_{1-4} indicating subsets of an event log L. The figure is adapted from Bose et al. [BvdAŽP11]. 52

3.2 Example of an extended prefix automaton for the event log $L = \{\langle a, b, c, d, f, e\rangle, \langle a, c, b, d, f, e\rangle, \langle a, c, b, d, e\rangle, \langle a, b, c, d, e\rangle\}$. The edges of the graph indicate activities, the nodes indicate states. A root node is added to mark the initial state. P_{1-4} refer to partitions of the automaton derived from the prefixes of the traces. The figure is adapted from Augusto et al. [AMVW22]. 56
3.3 Schematic depiction of evolutionary process mining based on [vdA16]. 60
3.4 (i) represents a (taxonomic) tree indicating the hierarchical relationships of the subgraphs n_{0-4}, (ii) represents the subgraph n_2, (iii) represents the subgraph n_3, and (iv) represents the subgraph n_4. The example is based on [GGP05]. . . 67
3.5 Proposed framework for the measurement of structural process variety. 69
3.6 An illustrative flat event log L' and the calculated distance matrix $D_{L'}$ based on Levenshtein distance. The example is taken from Bose and van der Aalst [JCBvdA10]. 71
3.7 The dendrogram derived from the hierarchical clustering of the trace variants in L′ and their pairwise distances $D_{L'}$. . . . 74
3.8 New distance matrix $D'_{L'}$ derived from the outcome of the hierarchical clustering. 76
3.9 A visualization of the decrease in variety according to the incremental reduction of trace variants from L'. The trace variants are removed based on their minimum distances. . . 77

4.1 Graph representations for the traces $\sigma_1 = \langle A, B, D, A, C, D, A, D\rangle$ (left) and $\sigma_2 = \langle W, X, Z, W, Y, Z, W, Z\rangle$ (right), which are discovered based on the directly follows relations between the activities. The colored edges indicate three different process patterns: loop (l), choice (c), and skip (s). 86

4.2 Different traces and their representation as trace graph. . . . 95
4.3 The tables show the calculated evaluation measures for the similarity measures on the basis of eight different event logs. The similarity measures proposed in Sect. 4.3 are marked in gray. Values that indicate best-performing similarity evaluations are marked in bold, performance improvement due to aggregations are underlined. 119

5.1 Trade-off in business process standardization between flexibility (measured in terms of structural process variety on the y-axis) and efficiency (measured in terms of the reduction of trace variants on the x-axis). The diagram also shows the contrary relation between standardization and adaptation costs. 128
5.2 The structural variety curve (blue) and the alignment costs curve (orange) in relation to the reduced number of trace variants (x-axis) and the structural log variety (y-axis). The diagrams show the results for the evaluation based on four different real-life logs, when ranking the trace variants based on (a) structural variety and (b) relative frequency. 135

6.1 Comparison between a local and global task based on a modularized process model consisting of three fragments. While both task types benefit from abstraction, the global task is additionally impeded by fragmentation, since the two relevant activities “A" and “B" are distributed among two fragments. 148
6.2 Example of a horizontally modularized process model following the fragment-based approach. 151

6.3 Distinction between search and inference phase. The blue dots and arrows depict a scan-path during search and the green dots and arrows depict a scan-path during inference. . 155
6.4 Research model for hypothesis testing. T - Theoretical construct, O - Operationalization of construct. 163
6.5 Four different control-flow patterns and how they are modeled for global tasks based on the fragment-based modeling approach. 166
6.6 User interface with an example of an experiment question. . 167
6.7 The different steps during each experiment session. 170
6.8 Approach for the segmentation and validation of the search and inference phases. 172

D.1 Process model fragments 1-4 as part of the fragment-based process model used to investigate local and global tasks types in the experiment. 266
D.2 Process model fragments 5-6 and three corresponding lifecycle models as part of the fragment-based process model used to investigate local and global tasks types in the experiment. 267

List of Tables

2.1 Example of a flat event log consisting of three trace variants. 11
2.2 Example for measuring the length of a trace. The comparison between empirical and formal relational system is based on Zuse [Zus91]. 22
2.3 Comparison between the causes, implications, and constraints of structural variety in automated workflows and evolutionary processes. 31

3.1 Overview of measures associated with process flexibility. . . 46
3.2 Overview of measures associated with process complexity. . 54
3.3 Overview of measures associated with exploratory processes. 58
3.4 Three artificial logs with distinct structural properties. 80
3.5 Comparison of different log variety measures (M.) based on seven different logs with low structural variety ($L_1 - L_3$), medium structural variety ($L_4 - L_6$), and high structural variety (L_7). Violations of monotonicity are marked in bold. 81

4.1 Overview of the desirable properties for similarity measures. 99
4.2 Selection of similarity measures. 100

4.3 Calculated similarity values for the pairwise comparison between the reference trace σ_0 and traces σ_3 through σ_8, with $s(\sigma_0, \sigma_3)$ serving as the baseline. According to the defined structural properties P6 to P8, specific expectations are associated with each comparison. The similarity value $s(\sigma_0, \sigma_4)$ is expected to reflect an increase in the number of activities, as defined by property P6. Similarly, $s(\sigma_0, \sigma_5)$ should capture an increase in the number of directly follows relations, in accordance with property P7. The comparisons $s(\sigma_0, \sigma_6)$ through $s(\sigma_0, \sigma_8)$ are intended to reflect an increase in the number of edge types, as specified by property P8. Any similarity value that is less than or equal to the baseline $s(\sigma_0, \sigma_3)$ indicates a violation of the corresponding property and is highlighted in gray to emphasize this deviation. . . . 101
4.4 Comparison between similarity measures based on the identified desirable properties P1-P4 and P6-P11. 102
4.5 Example for calculating the similarity between $\sigma_0 = \langle A, B, C, D \rangle$ and $\sigma_6 = \langle A, B, B, C, C, D \rangle$ based on the two measures Jg and Cg. 109
4.6 Set of proposed trace similarity measures. 109

4.7 Calculated similarity values for the pairwise comparison between the reference trace σ_0 and traces σ_3 through σ_8, with $s(\sigma_0, \sigma_3)$ serving as the baseline. According to the defined structural properties P6 to P8, specific expectations are associated with each comparison. The similarity value $s(\sigma_0, \sigma_4)$ is expected to reflect an increase in the number of activities, as defined by property P6. Similarly, $s(\sigma_0, \sigma_5)$ should capture an increase in the number of directly follows relations, in accordance with property P7. The comparisons $s(\sigma_0, \sigma_6)$ through $s(\sigma_0, \sigma_8)$ are intended to reflect an increase in the number of edge types, as specified by property P8. Any similarity value that is less than or equal to the baseline $s(\sigma_0, \sigma_3)$ indicates a violation of the corresponding property and is highlighted in gray to emphasize this deviation. . . . 111
4.8 Comparison between similarity measures based on the identified desirable properties P1-P4 and P6-P11. 111
4.9 Overview of the event logs including the number of activities, number of traces, the minimum and maximum length of traces, number of classes, number of traces per class, class type, and their origin. 114

5.1 Overview of the event logs including the number of activities, the number of trace variants, and their origin. 133
5.2 Comparison of the two trace variant selection approaches based on (a) structural variety and (b) frequency. 136

6.1 Demographic data of the participants. 168

6.2 Descriptive and inferential statistics showing the impact of the task locality on users' performance and cognitive load. N: number of observations, M: calculated mean, SD: calculated standard deviation. Units: Phase duration in seconds, Average fixation duration in milliseconds. A *p-value*$< .05$ means that the pairwise comparison results are significant, r^2 thereby indicates the effect size of the independent variable. 177
6.3 Descriptive and inferential statistics showing how users' search and inference phases differ between local and global tasks in terms of phase duration, cognitive load and scanpath variability. N: number of observations (cf. Sect. 6.3.3), M: calculated mean, SD: calculated standard deviation. Units: Phase duration in seconds, Average fixation duration in milliseconds. A *p-value*$< .05$ means that the pairwise comparison results are significant, r^2 thereby indicates the effect size of the independent variable. 178
6.4 Evaluation of the concept drift detection based on MAE for different similarity measures and window sizes (w). The values are separately calculated for global tasks (G.) and local tasks (L.). The values in brackets indicate the standard deviations for the calculated mean values. 185
6.5 Evaluation of the concept drift detection based on MSD for different similarity measures and window sizes (w). The values are seperately calculated for global tasks (G.) and local tasks (L.). The values in brackets indicate the standard deviations for the calculated mean values. 186

D.1 Eight process model comprehension tasks used to investigate local and global tasks types in the experiment. 265

A Axioms Related to Weitzman's Diversity Measure

The following axioms are identified by Bossert et al. [BPX03] and provide a description for the mathematical properties of the diversity measure $D_{\text{Weitzman}}(\Omega)$ proposed by Weitzman [Wei92].

Axiom 1 *(dissimilarity monotonicity). For any (not necessarily distinct) objects $w, x, y, z \in X$, $\{w, z\} \succeq \{x, y\}$ if and only if $d(w, z) \leq d(x, y)$.*

The axiom *dissimilarity monotonicity* states that the ranking of sets containing one or two objects should coincide with the ranking of the corresponding distances between the elements of those sets. It should be noted that the axiom does not require the objects $w, x, y, z \in X$ to be distinct.

Axiom 2 *(minimal distance independence). For any two sets A and B, and for any two objects x and y that do not belong to A and to B respectively, if the smallest distance between x and the other objects in A is smaller than the smallest distance between all distinct objects in A, if the same is true for y with respect to B and if the smallest distance between x and the other objects in A is the same as the smallest distance between y and the other objects in B, then $A \succeq B$ if and only if $A \cup \{x\} \succeq B \cup \{y\}$.*

The axiom *minimal distance independence* states that the diversity ranking of two sets should be independent of either adding or removing objects from the sets when the distances that separate the objects from the other objects in the sets fulfill the following conditions:

- the shortest distance between the added or removed object and any other object in the set is less than the shortest distance between any two objects already in the set
- the shortest distance between the object and any other object in the set is identical in both sets.

Axiom 3 *(link indifference). For all objects* $a, b, c, x, y \in X$ *such that* $x \neq y$ *and* a, b*, and* c *are pairwise distinct, if* $\{x, y\} \succ \{a, c\}$ *and if* b *is such that* $\max\{d(a, b), d(b, c)\} \leq d(a, c)$ *and* $d(a, c) + \min\{d(a, b), d(b, c)\} = d(x, y)$*, then* $\{x, y\} \sim \{a, b, c\}$.

The axiom *link indifference* states that two sets $\{x, y\} \succ \{a, c\}$ should have the same diversity if an additional object b is added to the lower-ranked set, such that:

- b lies in between a and c, i.e., the distances between b and a or c are less than the distance between a and c
- the resulting distance $d(a, c) + \min\{d(a, b), d(b, c)\} = d(x, y)$.

B Evaluation Measures for the Comparison of Trace Similarity Measures

The measures considered for the comparison of trace similarity measures in this thesis include *Nearest Neighbor*, *Precision@10*, and *Triplet*. These measures were originally proposed by Back et al. [BS23] for the evaluation of trace similarity measures and are well-established in research areas such as machine learning and data analytics.

Let $\phi : L \to X$ be a mapping of traces on a trace attribute $x \in X$. Let $L_x := \{\sigma \in L | \phi(\sigma) = x\}$ denote the set of traces with the attribute of value x.

The ***Nearest Neighbor*** evaluation measure calculates the expected ratio of traces whose nearest neighbor shares the same class label. The measure is defined as follows:

$$
\begin{aligned}
E_{NN}(L, d) &\equiv \frac{1}{|X|} \sum_{x \in X} \frac{1}{|L_x|} \sum_{\sigma_i \in L_x} \mathbf{1}(\sigma_i), \\
\mathbf{1}(\sigma_i) &= \begin{cases} 1 & \text{if } \phi\left(\arg\min_{\sigma_{j \neq i} \in L} d(\sigma_i, \sigma_j)\right) = \phi(\sigma_i) \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{B.1}
$$

The ***Precision@k*** evaluation measure captures the expected proportion of a trace's k nearest neighbors that share the same class label, denoted by $N_k(\sigma_i)$. The measure is defined as follows:

$$E_{\text{prec@}k}(L,d) \equiv \frac{1}{|X|}\sum_{x\in X}\frac{1}{|L_x|}\sum_{\sigma_i\in L_x}\frac{\{\sigma_j \in N_k(\sigma_i) \mid i \neq j \wedge \phi(\sigma_j) = \phi(\sigma_i)\}}{k} \tag{B.2}$$

The ***Triplet*** evaluation measure frames similarity metric error by asking whether a second data point of the same class is closer than a data point of a different class, given an anchor data point. Given a margin ϵ representing a minimum distance between classes, defined by $\max(d(\sigma_1,\sigma_2)+\epsilon-d(\sigma_1,\sigma_3),0)$, the triplet measure is defined as follows:

$$E_{\text{triplet}}(L,d) \equiv \frac{1}{|X|^2-|X|}\sum_{x,y\in X}\frac{1}{|L_x|}\sum_{\sigma_1\in L_x}\frac{1}{|L_x|}\sum_{\sigma_2\in L_x}\frac{1}{|L_y|}\sum_{\sigma_3\in L_y}\mathbf{1}(\sigma_1,\sigma_2,\sigma_3)$$
$$\text{where } x\neq y \text{ and } \mathbf{1}(\sigma_1,\sigma_2,\sigma_3) \equiv \begin{cases} 1 & \text{if } d(\sigma_1,\sigma_2)+\epsilon < d(\sigma_1,\sigma_3) \\ 0 & \text{if } d(\sigma_1,\sigma_2)+\epsilon \geq d(\sigma_1,\sigma_3) \end{cases} \tag{B.3}$$

A Python implementation of the evaluation measures can be found in the following repository:
https://github.com/promilab/StructuralProVar.git.

C Overview of Trace Similarity Measures

C.1 Process Flexibility Measures

Name	**Lempel-Ziv complexity**
Formula	$V_{LZ}(L) = \lvert\text{complexity}_{LZ}(L)\rvert$
Description	V_{LZ} equals the number of minimum unique subsequences derived by the Lempel-Ziv algorithm complexity_{LZ} [LZ76] in order to generate a sequence of of activities. To calculate V_{LZ} for event log L requires the concatenation of all traces in L.
Reference	[Pen03]
Name	**Deviation from random**
Formula	$V_{DR}(L) = 1 - (\sum_{i=1}^{\lvert A\rvert}\sum_{j=1}^{\lvert A\rvert}(\frac{m_{ij}-\bar{m}_{ij}}{\lvert F_{k=2}\rvert})^2)^{\frac{1}{2}}$
Description	V_{DR} equals the Euclidean distance [HKP12] between the observed process and a uniform random Markov process, where each activity is equally likely to occur independently from all previous activities [Pen03]. The distance is thereby calculated based on the respective relative frequencies m_{ij} of the observed directly follows relations $\langle a_i, a_j\rangle$ and an assumed constant relative frequency $\bar{m}_{ij} = \frac{\lvert F_{k=2}\rvert}{\lvert A\rvert^2}$, where $\lvert F_{k=2}\rvert$ is the total number of observed directly follows relations and $\lvert A\rvert$ the total number of activities.
Reference	[Pen03]
Name	**Average Levenshtein distance**

Formula	$V_{ALD}(L) = \frac{1}{\frac{1}{2}n(n-1)} \sum_{i=1}^{n} \sum_{j=i}^{n} d_{Lev}(\sigma_i, \sigma_j)$
Description	V_{ALD} equals the average pairwise distances between all traces in an event log $\sigma_i, \sigma_j \in L$. The distance measure d_{Lev} refers to the Levenshtein distance [Lev66] and $\frac{1}{2}n(n-1)$ reflects the number of pairs of traces.
Reference	[Pen03]

Name	**Number of events**
Formula	$V_{NE}(L) = \sum_{\sigma_i \in L} \lvert\sigma_i\rvert$
Description	Let $\lvert\sigma_i\rvert$ be the length of trace σ_i. V_{NE} equals the number of all events in L.
Reference	[Gün09]

Name	**Number of activities**
Formula	$V_{NA}(L) = \lvert A\rvert$
Description	Let A be a set activities. V_{NA} equals the number of all activities in L.
Reference	[Gün09]

Name	**Number of traces**
Formula	$V_{NT}(L) = \lvert C\rvert$
Description	Let C be a set of case ids. V_{NT} equals the number of all case ids (traces) in L.
Reference	[Gün09]

Name	**Activities per trace**
Formula	$V_{AT}(L) = \frac{\lvert A\rvert}{\lvert C\rvert}$
Description	V_{AT} equals the *number of activities* divided by the *number of traces*.

Reference	[Gün09]
Name	**Completeness of directly follows relations**
Formula	$V_{CDF}(L) = 1 - \frac{\lvert F_2 \rvert}{\lvert A \rvert^2}$
Description	Let F_2 be the set of all directly follows relations and A the set of all activities in an event log L. V_{CDF} equals the number of directly follows relations $\lvert F_2 \rvert$ divided by the maximum number of directly follows relations possible $\lvert A \rvert^2$.
Reference	[Gün09]
Name	**Similarity of directly follows relations**
Formula	$V_{SDF}(L) = \frac{\sum_{\sigma_i \in L} \sum_{\sigma_j \in L \setminus \sigma_i} d_J(\sigma_i, \sigma_j)}{n(n-1)}$
Description	V_{SDF} equals the mean Jaccard distance $d_J(\sigma_i, \sigma_j)$ between all traces in L. The Jaccard distance is calculated based on the overlap of directly follows relations (2-grams) observed in the traces.
Reference	[Gün09]
Name	**Trace entropy**
Formula	$V_{TE}(L) = -\sum_{\sigma_i \in L} p_i \log p_i$
Description	Let $a_i \in \mathbb{N}$ denote the number (absolute frequency) of trace variant $\sigma_i \in L$, such that $p_i = \frac{a_i}{\sum_{i=1}^{n} a_i}$ denote the proportion (relative frequency) of σ_i in L. $V_{TE}(L)$ is calculated as the Shannon-Wiener entropy over all traces variants in L.
Reference	[BDS19]
Name	**Prefix entropy**
Formula	$V_{PE}(L) = -\sum_{\lambda \in F_{\text{prefix}}} p_i \log p_i$

Description	Let F_{prefix} be the set of all prefixes observed in the event log L. Similarly to trace entropy V_{TE}, V_{PE} is calculated as the Shannon-Wiener entropy over all prefixes in L.
Reference	[BDS19]
Name	**Block entropy**
Formula	$V_{BE}(L) = -\sum_{\sigma_i \in F_k} p(\sigma_i) \log p(\sigma_i)$
Description	Let F_n be the set of all n-grams observed in the event log L. Similarly to trace entropy V_{TE}, V_{BE} is calculated as the Shannon-Wiener entropy over all n-grams in L.
Reference	[BDS19]
Name	**Entropy rate**
Formula	$V_{ER}(e_1, ..., e_n) = -\sum_{a_1 \in A} \cdots \sum_{a_n \in A} p_n \log p_n$
Description	Let $e_1, ..., e_n$ be discrete random variables and $a \in A$ the activities as possible outcomes. Furthermore, it is assumed that $p_n = p(e_1 = a_1, ..., e_n = a_n)$. V_{BE} is calculated as the Shannon-Wiener entropy over all possible outcomes of $e_1, ..., e_n$.
Reference	[BDS19]
Name	**Distance-based entropy**
Formula	$V_{DB}(L) = \frac{x}{n} \sum_{\sigma_i \in L'} \log d(\sigma_i, \sigma_{NN}) + \log \frac{\pi^{x/2}}{\Gamma(\frac{x}{2}+1)} + \gamma + \log(n-1)$

Description	V_{DB} is based on the Kozachenko-Leonenko entropy [DF17]. It considers the distance $d(\sigma_i, \sigma_{NN})$ between all traces and their nearest neighbor (NN), i.e., a distinct trace with minimum distance compared to all other traces in the event log L. Furthermore, x represents the underlying dimensionality of the metric space, $\Gamma(n)$ represents a generalization of $(n-1)!$ to real (and complex) numbers, and γ represents the Euler's constant.
Reference	[BDS19]

Name	**Relation entropy**
Formula	$V_{RE}(L) = -\sum_{a_i \in A}(\frac{c_A(a_i)}{\|A\|} log \frac{c_A(a_i)}{\|A\|} + \frac{c_N(a_i)}{\|A\|} log \frac{c_N(a_i)}{\|A\|} + \frac{c_S(a_i)}{\|A\|} log \frac{c_S(a_i)}{\|A\|})$
Description	V_{RE} equals the entropy over the *relation type count* for each activity $a_i \in A$. The relation type count is defined as a triple $(c_A(a_i), c_S(a_i), c_N(a_i))$ of numbers representing the activities, which follow after a_i in all traces (always), in a subset of the traces (sometimes), and in none of the traces (never). Originally, this measure is designed to derive a vector representing the relation entropy for all $a_i \in A$. However, to derive a single value the summation over all relation entropy values of the activities was added.
Reference	[BvdAŽP11]

Name	**Number of acyclic traces**
Formula	$V_{AT}(L) = \|F_{acyclic}\|$
Description	Let all be a set of acyclical traces, i.e., traces without a loop. V_{AT} equals the number of acyclic traces contained in L.
Reference	[PLKH20]

C.2 Process Complexity Measures

Name	**Events per trace**
Formula	$V_{ET}(L) = \frac{\sum_{\sigma_i \in L} \lvert\sigma_i\rvert}{\lvert C\rvert}$
Description	V_{ET} equals the number of all events in an event log L divided by the number of all traces in L.
Reference	[BWBBG14]
Name	**Activities per trace**
Formula	$V_{AT}(L) = \frac{\lvert A\rvert}{\lvert C\rvert}$
Description	V_{AT} equals the number of all activities in an event log L divided by the number of all traces in L.
Reference	[BWBBG14]
Name	**Activity event ratio**
Formula	$V_{AET}(L) = \frac{V_{AT}(L)}{V_{ET}(L)}$
Description	V_{AET} relates the average number of activities over all traces to the average number of events over all traces.
Reference	[BWBBG14]
Name	**Activity per trace ratio**
Formula	$V_{ATR}(L) = 1 - \frac{V_{AT}(L)}{\lvert A\rvert}$
Description	V_{ATR} considers the overall probability of whether a particular activity can be observed in any trace by relating the average activities per trace to the total number of activities.
Reference	[BWBBG14]
Name	**Modified average Levenshtein distance**

Formula	$V_{MALD}(L) = \frac{1}{\frac{1}{2} * n * (n-1) * V_{ET}(L)} \sum_{i=1}^{n} \sum_{j=i}^{n} d_{Lev}(\sigma_i, \sigma_j)$
Description	$V_{MALD}(L)$ is an extended version of the average Levenshtein distance $V_{ALD}(L)$, which also controls for the differences in trace lengths based on the events per trace $V_{ET}(L)$.
Reference	[BWBBG14]

Name	**Variant entropy**
Formula	$V_{VE}(L) = \lvert S \rvert \log(\lvert S \rvert) - \sum_{i=1}^{max(C)} \lvert \{s \in S \lvert C(s) = i\} \rvert \log(\lvert \{s \in S \lvert C(s) = i\} \rvert)$
Description	Let S be the set of states in an extended prefix automata, and $C(s)$ a function partitioning for each state $s \in S$. V_{VE} equals the Shannon-Wiener entropy over all partitions of an extended prefix automata.
Reference	[AMVW22]

Name	**Normalized variant entropy**
Formula	$V_{NVE}(L) = \frac{V_{VE}}{\lvert S \rvert \log(\lvert S \rvert)}$
Description	V_{NVE} is a normalized version of the variant entropy V_{VE}, considering all observed states in an extended prefix automata.
Reference	[AMVW22]

Name	**Sequence entropy**
Formula	$V_{SE}(L) = \lvert seq(S) \rvert \log(\lvert seq(S) \rvert) - \sum_{i=1}^{max(C)} \lvert seq_i(S) \rvert \log(\lvert seq_i(S) \rvert)$

Description	Let S be the set of states in an extended prefix automata, $C(s)$ a function partitioning for each state $s \in S$, and $seq(S)$ a function mapping all states $s \in S$ to a set of events. V_{SE} equals the Shannon-Wiener entropy over the partitions and the events per state of an extended prefix automata.
Reference	[AMVW22]

Name	**Normalized sequence entropy**
Formula	$V_{NSE} = \frac{V_{SE}}{\lvert seq(S)\rvert \log(\lvert seq(S)\rvert)}$
Description	V_{NSE} is a normalized version of of the sequence entropy V_{SE}, considering all observed events per states in an extended prefix automata.
Reference	[AMVW22]

C.3 Exploratory Process Measures

Name	**Gini coefficient**
Formula	$V_{GC}(L) = 1 - 2\int_0^1 L(x)dx$
Description	Let $L(x)$ be the Lorenz curve denoting the percentage traces generated by the lowest 100x% of activities during a fixed time period. V_{GC} measures the distributional inequality of activities within an event log L.
Reference	[FH07]

Name	**Average distance**
Formula	$V_{AD}(L) = \frac{1}{\frac{1}{2}n(n-1)} \sum_{i=1}^{n} \sum_{j=i}^{n} d(\sigma_i, \sigma_j)$
Description	V_{AD} is the average distance between all pairs of traces in an event log L based on some distance measure $d(\sigma_i, \sigma_j)$.

Reference	[CBB13, LCB14]
Name	**Distance to average point**
Formula	$V_{DAP}(L) = (\sum_{j=i}^{n}(\overrightarrow{\sigma_i} - \overrightarrow{\sigma})^2)^{\frac{1}{2}}$
Description	Let $\overrightarrow{\sigma}$ denote an average trace derived through the mapping of all traces in L into some vector space. V_{DAP} equals the Euclidean distance over all traces and the average trace.
Reference	[Urs02]
Name	**Average Manhattan distance**
Formula	$V_{AMD}(L) = \sum_{i=1}^{n} \|\|\overrightarrow{\sigma_i} - \overrightarrow{\sigma}\|\|$
Description	V_{AMD} is calculated as the average Manhattan distance between a vector representation of all directly follow relations observed in σ_i and the average frequency of directly follows relations observed over all $\overrightarrow{\sigma} = \frac{\sum_{i=1}^{n} \overrightarrow{\sigma_i}}{\|C\|}$
Reference	[MRD+15]

D Experiment Material

Type	Control-flow	Task
Local	Ordering	The activity "Enter container information from documents" needs to be executed before the activity "Create new erp system entry"?
	Concurrency	The activity "Inform customs about temporary storage" can be executed in parallel with the activity "Inform shipping company about temporary storage"?
	Exclusiveness	The activities "Send detailed information on final destination" and "Load container onto train" are mutually exclusive?
	Repetition	The activities "Request independent analysis of damage" and "Investigate cause of shake" can be executed several times?
Global	Ordering	The activity "Drive container to scanning platform" needs to be executed before the activity "Attach accelerometer sensor"?
	Concurrency	The activity "Document sensor id and container id" can be executed in parallel with the activity "Calibrate sensor"?
	Exclusiveness	The activities "Request container deletion" and "Schedule transportation from temporary storage" are mutually exclusive?
	Repetition	The activities "Document results of scan in erp file" and "Log into erp system" can be executed several times?

Table D.1: Eight process model comprehension tasks used to investigate local and global tasks types in the experiment.

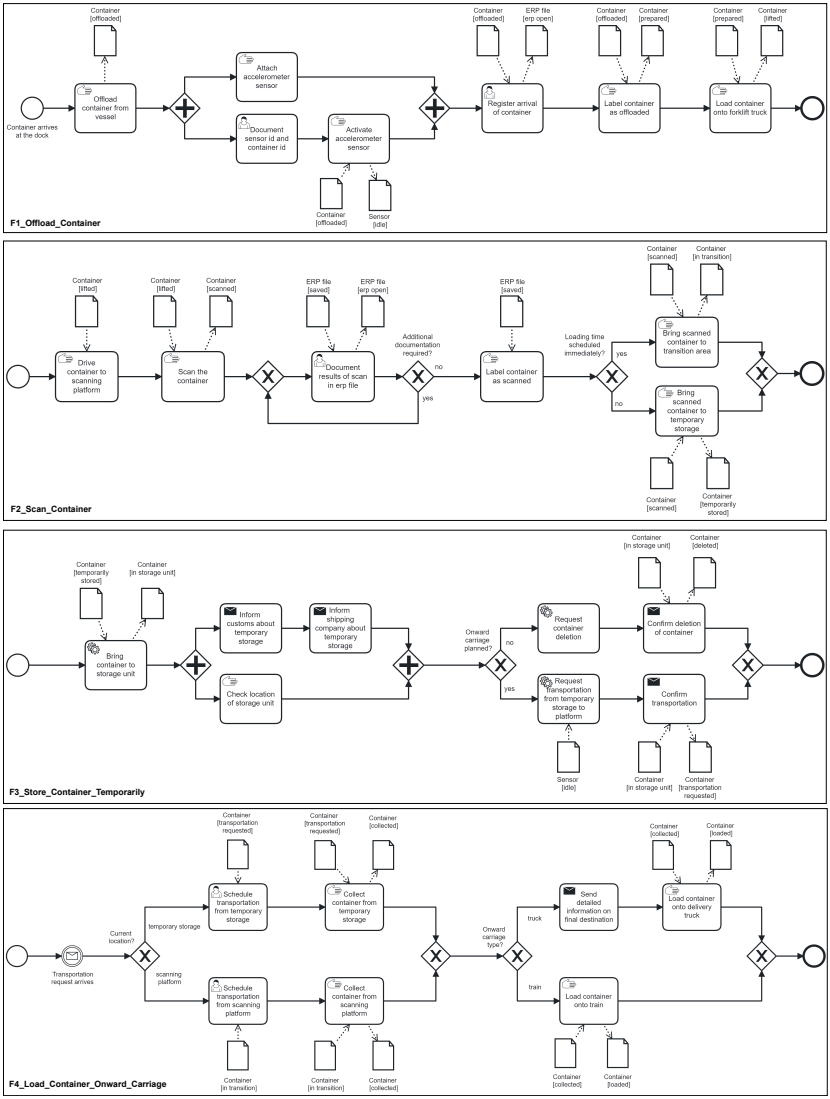


Figure D.1: Process model fragments 1-4 as part of the fragment-based process model used to investigate local and global tasks types in the experiment.

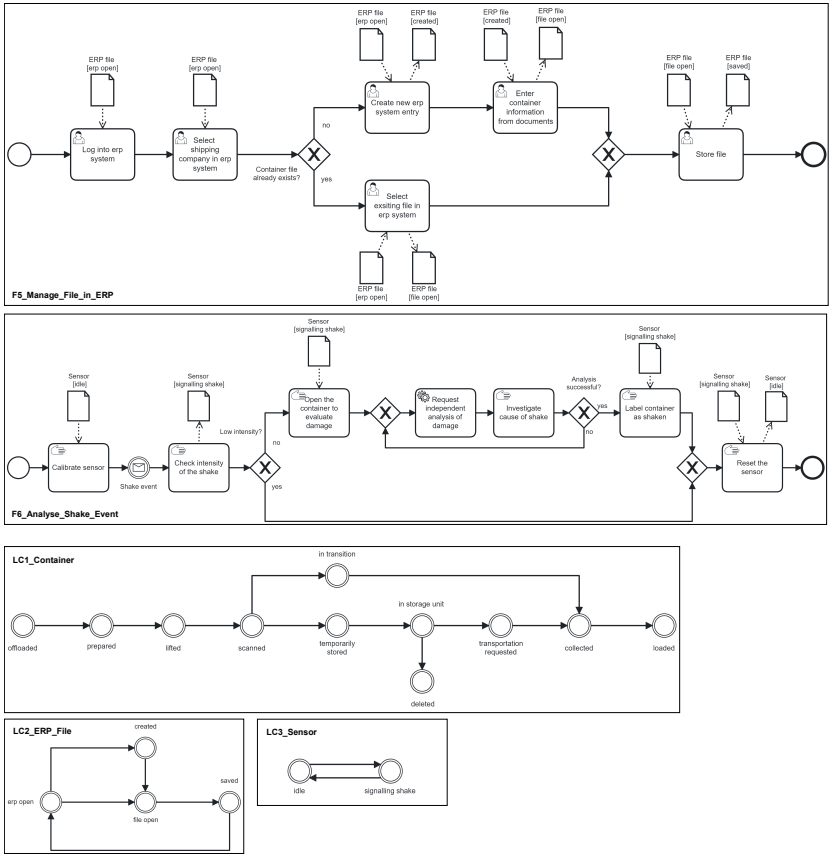


Figure D.2: Process model fragments 5-6 and three corresponding life-cycle models as part of the fragment-based process model used to investigate local and global tasks types in the experiment.