UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Software Engineering Curriculum

Serhii Murashko

Discovery and Simulation of Business Process with Multiple Data Attributes and Conditions

Master's Thesis (30 ECTS)

Supervisor(s): Orlenys López-Pintado, PhD
Marlon Dumas, Professor

Tartu 2024

Discovery and Simulation of Business Process with Multiple Data Attributes and Conditions

Abstract: Business process simulation (BPS) is a crucial tool for organizations, allowing them to forecast outcomes and assess the impact of potential changes within their processes. This capability supports effective decision-making by enabling "what-if" scenario analysis. However, traditional BPS models rely on a limited number of attributes (activity names, resources, and timestamps) with probabilistic decision-making, overlooking process-dependent data attributes. For example, in emergency services or patient care processes, using probabilistic decisions might miss important details about patient conditions or available resources, which could trigger critical issues.

This thesis introduces a Data-Aware Simulation (DAS) model, designed to incorporate dynamic attributes and enable data-aware decisions within simulations. The DAS model addresses the limitations of traditional approaches categorising attributes into 3 types, case, global, and event, to cover different scopes (local or global) and behaviour (static or dynamic) of the attributes. These attributes provide branching conditions at the decision points based on the current data state to guide the execution flow of the process.

Another significant aspect of this research is the discovery of the DAS model from the event logs. By incorporating the DAS model with discovery tools, organisations can discover their simulation models, including the perspective of data attributes and branching conditions that often affect the execution flow of business processes. Following discovery, organisations can simulate the model and make necessary adjustments and optimisations to adapt the models to reflect changes in operational procedures or to explore different 'what-if' scenarios, thereby maintaining their relevance and effectiveness in dynamic business environments.

The evaluation demonstrates that data-aware models, discovered from event logs, can accurately classify data attributes, their update mechanisms, and implications into branching conditions. These models also replicate the control flow of the original log while enhancing cycle and event times, in contrast to traditional non-data-aware models that depend on branching probabilities.

Keywords: Business Process Simulation, Business Process Discovery, Data-Aware Simulation Model, Simulation Data Attributes, Data-Aware Decision Making

CERCS: P170 - Computer science, numerical analysis, systems, control

Äriprotsessi Avastamine ja Simuleerimine Mitme Andmeatribuudi ja Tingimustega

Lühikokkuvõte:

Äriprotsesside simulatsioon (APS) on organisatsioonide jaoks oluline vahend, mis võimaldab neil prognoosida tulemusi ja hinnata võimalike muutuste mõju oma protsessides. See võime toetab tõhusat otsuste tegemist, võimaldades stsenaariumide analüüsi. Traditsioonilised APS-mudelid põhinevad siiski piiratud arvul atribuutidel (aktiivsusnimed, ressursid ja ajatemplid), millel on tõenäosuslik otsuste tegemine, jättes kõrvale protsessist sõltuvad andmeatribuudid. Näiteks kiirabiasutustes või patsientide raviprotsessides võib tõenäosuslike otsuste tegemisel jääda vajaka olulistest üksikasjadest patsientide seisundite või olemasolevate ressursside kohta, mis võivad esile kutsuda kriitilisi küsimusi.

Käesolev töö tutvustab Andmeteadlikku Simulatsioonimudelit (ASM) mudelit, mille eesmärk on kaasata dünaamilisi atribuute ja võimaldada andmeteadlikke otsuseid simulatsioonides. ASMi mudel käsitleb traditsiooniliste lähenemisviiside piiranguid, mis liigitavad atribuudid 3 tüübiks, juhtumiks, globaalseks ja sündmuseks, et hõlmata atribuutide erinevaid mõõtkavasid (kohalik või globaalne) ja käitumist (staatiline või dünaamiline). Need atribuudid pakuvad otsustuspunktides hargnemistingimusi, mis põhinevad praegusel andmete seisundil, et suunata protsessi täitmise voogu.

Teine oluline aspekt selles uurimistöös on ASM mudeli avastamine sündmuste logidest. Lisades ASMi mudeli koos avastamisvahenditega, saavad organisatsioonid avastada oma simulatsioonimudeleid, sealhulgas andmete atribuutide perspektiivi ja hargnemistingimusi, mis mõjutavad sageli äriprotsesside teostusvoogu. Pärast avastamist võivad organisatsioonid mudelit simuleerida ning teha vajalikke kohandusi ja optimeerimisi, et kohandada mudeleid nii, et need kajastaksid muudatusi töökorras või uuriksid erinevaid "mis-kui-stsenaariume", säilitades seeläbi nende asjakohasuse ja tõhususe dünaamilises ärikeskkonnas.

Hindamine näitab, et sündmuste logidest avastatud andmeteadlikud mudelid võivad täpselt liigitada andmeatribuudid, nende ajakohastamismehhanismid ja mõju hargnemistingimustesse. Need mudelid kopeerivad ka algse logi kontrollvoolu, suurendades samal ajal tsükli ja sündmuste aegu, erinevalt traditsioonilistest mitteandmetest teadlikest mudelitest, mis sõltuvad hargnevatest tõenäosustest.

Võtmesõnad:

Äriprotsessi Simulatsioon, Äriprotsessi Avastamine, Andmeteadlik Simulatsioonimudel, Simulatsiooni Andmeatribuut, Andmeteadlik Otsustamine

CERCS: P170 - Arvutiteadus, arvanalüüs, süsteemid, kontroll

Contents

1 Introduction

Business Process Simulation (BPS) is a technique that enables companies to evaluate the potential impact of changes in their business processes. By simulating changes, organizations can conduct *"what-if"* analysis and observe the outcomes. For example, BPS can help answer questions like, "What if we extend the operating hours of a production line by two hours daily?" This approach allows organizations to test the potential effects of decisions before implementing them in real life.

BPS involves two main parts: the *BPS model* and the *BPS engine*. The BPS model includes *control flow*, which is described using *Business Process Model and Notation* (BPMN) [1] and *simulation scenario*. BPMN organises the sequence of activities and events in the business process.

In BPMN, an *activity* is a piece of work that needs to be done, such as checking the inventory or approving a request. An *event* is something that happens instantly, such as receiving an email or an alarm that goes off. These activities and events are connected by *sequence flows (arcs)*, which show the order in which they occur. *Gateways* within these flows direct the process path: an *XOR gateway* directs the flow to precisely one of many branches based on probability, an *OR gateway* allows one or more branches to be taken, and an *AND gateway* enables all outgoing branches to proceed concurrently.

The components of the simulation scenario can vary depending on the implementation of the simulation engine. Standard sections usually include: *Scenario Specification*, determines the number of process instances to generate and start date and time of the process; *Arrival Calendar*, dates and times when cases are expected to arrive; *Arrival Rate*, the frequency at which new cases appear; *Resource Calendars*, dates and times when resources are available; *Resource Profiles*, details about the resources such as number, cost, and schedule; *Resource Allocation*, how resources are distributed to tasks; *Branching Probabilities*, the chances of each flow being chosen at decision points (gateways); and *data attributes*, which store additional data about the changing values of attributes during the simulation.

Key Performance Indicators (KPIs) are metrics used to measure the efficiency of a business process managed through BPS, and these metrics can vary based on the simulation model's implementation. These include *Waiting Time*, the time from when an activity is enabled until it starts; *Processing Time*, the duration from start to finish of an activity; *Cycle Time*, the total time from the start of the first activity to the end of the last activity in a process; and *Resource Utilization*, how much of the available time resources are used [2].

BPS model discovery translates raw event log data into a simulation model by analysing patterns and correlations in the data representing organisational operations. Discovery may cover only simulation scenarios with all components described before if control flow has been provided or both. The extraction of a ready-to-use simulation model allows organisations to simulate and analyse their business processes to predict outcomes under various scenarios, thereby supporting decision-making and optimization [3]

Employing BPS discovery is crucial for organisations that lack a predefined model of their operations because they can reconstruct their actual workflows, identify inefficiencies, and explore potential improvements without a clear understanding of the underlying business processes. For example, when an organisation might only have access to logs generated by their IT systems.

Existing simulation models focus primarily on static event log attributes such as case IDs, activity names, resources, and timestamps, which remain consistent across different processes. This approach often neglects the dynamic (process-dependent) data attributes specific to each process and can vary significantly. The previous model [4] initially attempted to address this problem by introducing *case_attributes*, which represent data attributes initialised at the start of a case and remain unchanged throughout the simulation. These attributes are not influenced by any activity and are isolated to their specific cases. For example, in Table 1, beyond common attributes, an event log could include additional columns depending on the business process, such as the loan amount (LA) and the type of client (CT) in a **loan application** or hospital capacity (HC) and assigned doctor (AD) in **hospital**.

Table 1. Examples of Event Logs with Different Attribute Levels

Common Attributes					Loan Application		Hospital	
Case	**Activity**	**Resource**	**Start**	**End**	**LA**	**CT**	**HC**	**AD**
1	A	Res 1	8:00	8:30	500	Regular	20	Dr. Smith
2	B	Res 2	9:00	9:45	1000	Premium	18	Dr. Jones
3	C	Res 3	7:00	7:20	700	Regular	19	Dr. White

However, this approach does not consider dynamic changes during case execution. Consider a hospital scenario where the number of available beds is established at the beginning of the day but needs to be updated as patients are admitted and discharged. The static nature of *case attributes* does not capture such dynamic changes. As illustrated in the referenced figure, only one out of four possible data behaviours are covered by the existing model, focussing solely on static, local attributes. This leaves gaps in capturing local dynamic attributes, global static attributes, and global dynamic attributes described in Table 2. To bridge these gaps, we can introduce a global scope in which attributes are shared across all cases. By establishing update rules, we can dynamically calculate new values based on the current states of other data attributes within the simulation model.

Table 2. Attribute scopes in business process simulations.

	Static	**Dynamic**
Local	Case Attributes	Event Attributes
Global	Global Fixed Attributes	Global Attributes

Another limitation of traditional models is their reliance on predefined probabilities at decision points (gateways), which do not consider actual case data. For example, in a loan application process, decisions to approve or decline a loan might be made without considering relevant case specifics. To address this, we can utilise the previously mentioned dynamic and global data attributes, setting conditions at decision points that influence both the data flow and control flow within the business process. This allows for a more nuanced approach in which decisions, such as performing a detailed check for regular customers requesting loans over $10,000, are based on real-time data. Implementing data-aware conditions at decision points improves the accuracy and relevance of simulations, ensuring that the business process results reflect the actual operational scenarios more accurately [5].

To address the challenges in the simulation model and ensure a user-friendly workflow in which organisations can easily discover and simulate their process models from event logs, we established the following Research Goals (RGs):

- **RG1**: Design a model capable of handling data attributes and data-aware decision-making.
- **RG2**: Discover the model described in **RG1** from event logs, focusing on capturing branching conditions and data attributes.
- **RG3**: Enable the simulation of the model designed in **RG1** to assess and validate its dynamics and decision-making capabilities.

We will utilise a specific simulation engine called Prosimos [4] and a Simod discovery tool [6, 7] to achieve these goals. Although each tool can function independently, they share compatible data structures facilitating their integration. This compatibility allows for the seamless integration of the DAS model, with only minor isolated modifications needed for each tool.

In projects, we use the Systems Development Lifecycle (SDLC) to ensure that our work aligns with the project's needs. The SDLC framework allows us to choose and adjust different stages as needed, following the guidelines of the ISO/IEC/IEEE 12207 standard[1]. This approach allows us to manage the projects in flexible stages rather than in a fixed sequence. We cover all phases of SDLC: requirement gathering, analysis, design, development, testing, deployment, and maintenance.

For development, we follow Agile methods, specifically the Scrum framework [8]. This process starts with setting the initial requirements and continues through implementation, testing, and maintenance cycles. This cycle helps us adapt and improve the projects as they progress [1].

During thesis writing, ChatGPT-4[2] and DeepSeek Coder[3] were employed to debug and restructure the project code and explain complex terms and algorithms in understandable language. In addition, we used ChatGPT-4 and Grammarly[4] to review the written text and improve the clarity and grammatical precision of the thesis. Based on feedback and recommendations received, the text of the thesis was adjusted and spelling errors were corrected.

The rest of the thesis is structured as follows. Section 2 covers background information and related works, discussing essential concepts and summarising what has already been done before. Section 3 describes the DAS model and its components with a data-aware approach from the simulation and discovery perspectives, respectively. In Section 4, we refer to implementation and explain in detail the testing phase in the simulation tool, setting up experiments, and evaluating the results of the DAS model compared to the traditional one. Section 5 summarises the work completed and explores potential improvements.

[1] https://www.iso.org/standard/63712.html
[2] https://chatgpt.com
[3] https://chat.deepseek.com/coder
[4] https://app.grammarly.com/

2 Background and Related Work

2.1 Business Process Management

Business Process Management (BPM) combines principles, methods, techniques, and tools to monitor and improve the life cycles of business processes. According to [9], the life cycle of a business process includes stages such as identification, discovery, analysis, redesign, implementation, execution, monitoring, and adaptation. BPS is a key technique used in the analysis and discovery stages. It allows the simulation of business processes to gather important numerical data, such as performance metrics, such as cycle time and processing time, and assess potential changes and their impacts.

We use a discovery tool to identify input parameters for the simulation engine. The inputs for this tool include a business process model and an event log file. An event log captures each instance of the business process, termed a "case." Each case within the log details various attributes: *case identifier* (a unique ID for each case), *event name* (the name of the task or event), *resource* (the entity that executes the task), and timestamps such as *enabled time*, *start time*, and *end time*, which indicate when a task was ready to start, began, and completed, respectively. All previously mentioned attributes will be called *fixed* or *common* attributes because they are present in every event log, regardless of their specific details. Attributes that go beyond the common are called data attributes and contain additional process-dependent information.

In our approach, we specifically examine the data attributes captured in the event log. For example, this may include information like loan amounts, application statuses, or any other specific data recorded during process execution. This allows us to enrich the analysis with a broader range of data and support data-aware BPS.

2.2 Business Process Management and Notation

In Business Process Management (BPM), it is crucial to present business processes clearly so everyone involved understands them. We use the Business Process Model and Notation (BPMN) [1], a widely accepted standard, to create business process diagrams that are easy to interpret. BPMN's graphical elements help visualise the interactions within business processes.

BPMN includes several key components:

- **Activity**: Represents a task within the process performed by a resource such as a person or a machine. Activities push the process forward by completing the required work.

- **Event**: Marks a specific occurrence within the process, occurring instantly. Events in BPMN are categorised into three types: start, intermediate, and end, corresponding to their roles at various stages of the process. For example, receiving a loan application form can be a start event, initiating the process, while receiving a loan rejection letter might represent an end event, signalling the process's conclusion.

- **Gateway**: Serves as a decision point that influences the direction of the process flow. The gateways determine which path to follow based on the probabilities or conditions, if any, at that point.

- **Arc (Sequence Flow)**: These lines connect the elements and show the order in which activities, events, and decisions occur within the process.

Since our focus is not on events defined in BPMN, we will use the term *event* to refer to a specific activity recorded in the event log. This includes all associated attributes, such as timestamps, resources, and data attributes, that represent a single event log row.

To understand how business processes are managed in BPMN, we use the concept of a *token*, which helps to track the progress of each process instance. According to [9], tokens represent the current state and move through the process as different events and activities occur. When a process starts with a triggering event, a token is placed on the outgoing arc of that event. This placement signals that the next element in the sequence flow is ready to begin. If this next element is an activity, it becomes *enabled*—meaning the token reaches it and waits for a resource to become available to start the activity. The activity begins once a resource starts working on it, moving the token into the activity. Upon completion of the activity, the token moves to the following sequence flow, indicating the end of one task and the readiness to start another. This movement continues until there are no more tokens in the process, indicating the end of the process instance. Unlike activities, events do not hold tokens for any duration; they occur instantly and move the token immediately to the next element.

Among the types of gateways, the following are particularly significant.

- **Exclusive (XOR) Gateway**: This type allows only one of the possible paths to be followed based on a specific probability or condition associated with each path. As a result, only 1 token will be produced.

- **Parallel (AND) Gateway**: Enables multiple paths to be executed simultaneously, regardless of probabilities or condition evaluation, and creates a token for each executed path.
- **Inclusive (OR) Gateway**: Allows one or more paths to be activated based on probabilities or evaluated conditions, allowing multiple process branches to run concurrently. Creates one or more tokens, depending on the path selection.

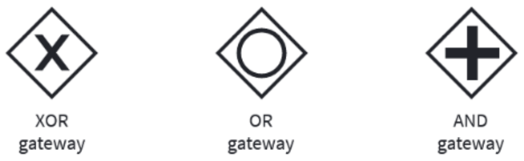


Figure 1. BPMN gateway types

This thesis will concentrate on XOR and OR gateways. Unlike AND gateways, which operate all connected paths simultaneously, XOR and OR gateways may use dynamic decision-making based on attributes and data-aware logic instead of relying on probabilities.

2.3 Business Process Simulation

Improving business processes in the real world can be challenging, costly, and time-consuming. Simulation offers a method to test various improvement strategies without directly implementing them. This approach allows for the setting of the desired number of cases within a simulation scenario, case execution, and quantitative analysis of the results. Afterwards, adjustments can be made based on simulation results to further optimise the process. Our simulations are performed using *Prosimos* [4], which incorporates the Business Process Model and Notation (BPMN) for defining process models.

Although various simulation tools are available, each implements different strategies to handle attributes and decision-making in simulations. Some tools offer a more flexible and configurable approach across multiple levels of a process, while others may provide less flexibility but enhance other aspects of simulation. We have selected two simulation tools, *iGrafx* and *Apromore*, to investigate their approaches to data and data-aware decision-making in process modelling.

iGrafx iGrafx [5] allows the assignment of attribute values at several key points during the execution of a task within a process model. These points include the entry into the task after the input has been received before the task starts after the task completes, and as the process exits the task. This setup provides the flexibility to modify the attributes that influence the process at multiple stages. Variables in iGrafx can be scoped globally, affecting the entire model, or locally, impacting only specific cases. In addition, iGrafx gateways can be based on conditions or set probabilities. However, iGrafx can be complex to set up and might require a steep learning curve for new users, with potential difficulties in managing global variables, debugging complex models, and the need for precise configuration and extensive testing.

Apromore Apromore [6] distinguishes between the case and event attributes within its simulation framework. The case attributes remain constant throughout the lifecycle of a case, while the event attributes can be modified by task executions. However, Apromore primarily uses generators for these modifications, meaning that the variables do not depend on their previous values but are instead regenerated based on specified distribution functions at each task execution. While this simplifies some modelling aspects, it can also limit the tool's ability to simulate more complex, interdependent scenarios.

Going deeper into our model, the DAS model in Prosimos includes several components that define the operational parameters, where *Event Attributes*, *Global Attributes*, *Gateway Branching Conditions* components are extensions of the thesis.

- **Scenario Specification**: Specifies the number of process instances to be simulated and the simulation's start date and time.
- **Arrival Calendar**: Defines specific time intervals during which new process instances can start, where each interval specifies a start and end time. For example, from 7 AM to 6 PM on weekdays.
- **Arrival Rate**: Details how frequently new instances appear, using distribution functions to model this variability. For instance, instances might arrive following a normal distribution with a specified mean and variance.

[5] `https://www.igrafx.com`
[6] `https://apromore.org`

- **Resource Calendars**: Shows when resources, such as personnel or equipment, are available to perform tasks, including time windows for resource utilization.
- **Resource Profiles**: Describes the resources available for process tasks, including details like identifiers, cost per hour, and availability, categorized into different resource pools.
- **Resource Allocation**: Assigns specific tasks to resources based on their capabilities and availability, considering scenarios where multiple resources can perform the same task.
- **Case Attributes**: Initialized at the creation of each case, values set by generators that remain constant throughout the case lifecycle.
- **Event Attributes**: Modified by activities during the process execution, changes are local to the specific case.
- **Global Attributes**: Similar to event attributes, the value is shared across all cases, reflecting changes that affect the entire simulation environment.
- **Branching Conditions**: A set of rules that evaluate the current state of data attributes (case, event, and global) to define conditions that influence the control-flow at gateways.
- **Gateway Branching Probabilities**: Manages decision points within the process by assigning a probability to each pathway at the gateways. Each path can also have a reference to specific branching conditions from *branching conditions* component. Probabilities are used only after the branching conditions have been assessed and are necessary to determine the path forward. Probabilities range from 0 to 1, with the sum of all probabilities for a single gateway always equaling 1.

With this approach, the DAS model can now simulate varied behaviours from a data perspective, allowing flexible configuration of the simulation model. This capability enables a more precise and dynamic simulation of real-life processes. For example, a *Case Attribute* might be the name of a financial institution in a financial management simulation, established at the beginning and maintained constant throughout a specific case. An *Event Attribute* could be the count of customer interactions handled by a service agent, updated each time a new interaction is recorded. A *global attribute* could represent the total sales volume in all cases, dynamically updated as transactions occur in any of the process instances.

Branching conditions, by using simulation data, provide the ability to include a dynamic data-driven approach in decision-making. This approach uses the values and dynamic behaviours of the simulation data to make real-life decisions based on actual data rather than relying fully on probabilities at the decision points. However, the model can still use probabilities as a backup plan if the conditions do not work as expected, ensuring that the simulation progresses without interruption.

For example, an XOR gateway directs passengers to different security checks based on their status at an airport. Premium passengers use an expedited check, and regular passengers pass a standard check. If a passenger is mistakenly registered as "staff," which does not match any conditions for the available paths, they would typically be stuck. To prevent flow interruption, the system then applies predetermined probabilities instead of conditions to direct the passenger.

2.4 Business Process Simulation Model Discovery

The discovery of the BPS model is important for turning data from event logs into simulation-ready models. This process helps to connect the theoretical ideas of business models with the actual operations of real-world processes. We use tools like *Simod*, which automatically configure and discover BPS models, to effectively analyse event logs [6]. These tools help us develop detailed simulation models that include control flows, resource allocations, and the timing of activities - important parts that reflect actual business process actions [3].

Event logs usually record data like *case_id*, *resources*, *activity names*, and the *start* and *end* times. This information is the main input for *Simod*, helping to discover all the necessary components to create a precise simulation scenario [10]. This makes simulation of business processes possible in varied operational settings [5]. Automating this discovery process not only boosts the accuracy of the simulation models but also cuts down the time and effort required for model development. This capability is highly beneficial for organisations that seek to analyse and improve their operational processes efficiently [11].

Dynamic Attribute Discovery In dynamic attribute discovery, data not categorised under traditional labels like *case_id*, *resources*, *activity names* or timestamps (*start time* and *end time*) is considered *data attributes*. These could include any operational data captured in the event log. The discovery process involves identifying patterns and the scope of these attributes to determine their behaviour and classification as case, global, or event attributes.

Moreover, global and event attributes can be manipulated by updating rules beyond basic distribution functions. These rules might include mathematical operations or functions that dynamically change the attributes' values. For example, such rules could modify the *loan_amount* by recalculating it as *loan_amount* = *loan_amount* * *rate*, or they might increment the *customers_served* count by updating it with *customers_served* = *customers_served* + *1* for each new customer interaction.

For example, in Table 3 you can observe how the case, event, and global attributes behave within a simulated business process. The *caseAttr* is initialised using a random value from 1 to 10. Update rules for *eventAttr* and *globalAttr* are applied as follows: activity A adds 1, B adds 5, and C multiplies the value by 2. However, the results differ due to the attributes' scope. Specifically, the *eventAttr* uses its most recent value from the same case to update, while the *globalAttr* updates based on the latest value from the entire simulation, including all cases.

Table 3. Example of case, event, and global attributes within event log.

Case ID	**Activity**	**caseAttr**	**eventAttr**	**globalAttr**
1	A	5	1	1
1	B	5	6	6
1	C	5	12	12
2	A	7	1	13
2	B	7	6	18
2	C	7	12	36
3	A	2	1	37
3	B	2	6	42
3	C	2	12	84

Data-Aware Decision Making Discovery For data-aware decision-making, we focus on discovering branching rules that are influenced by attribute states at decision points. This involves replaying the event log to observe simulation states at the decision points, allowing us to identify how different attributes influence pathway selections at gateways. For example, a gateway may direct cases differently based on whether a customer is marked as 'premium' or 'ordinary.' However, in addition to conditions, probabilities are calculated based on frequencies and historical data.

Thus, it ensures that the simulation model continues even when unexpected data scenarios arise, such as all conditions at a gateway being false or the model being unable to discover the rule due to its complexity. In such cases, the model uses predefined probabilities to maintain continuity and prevent process disruption.

3 Data-Aware Simulation Model and Discovery

In this section, we discuss the DAS model and its discovery. Section 3.1 explains how the model incorporates dynamic attributes and data-driven decision-making into business process simulations. Section 3.2 outlines the discovery process of the DAS model, beginning with the classification of simulation data attributes and the discovery of update rules. The section concludes by describing the discovery of branching conditions and discussing methods to analyse and capture the decision-making logic at decision points.

3.1 Data-Aware Simulation Model

The DAS model, detailed in Definition 3, is based on the probabilistic model [4] that used PDM, explained in Definition 2, for decision-making in gateways, introduced the first data attribute, namely *case_attributes*, to support process prioritisation [12] and the concept of *generators* explained in Definition 1.

Definition 1 (Generators (GN))**.** *GN creates new values for attributes without considering previous values. There are two main types of generators:*

- ***Continuous Values****: Use distribution functions to produce numerical values. The number of parameters needed varies depending on the used function (fixed, uniform, normal, exponential distributions, etc.)*
- ***Discrete Values****: Create categorical values based on set probabilities. Each value is required to have a probability between 0 and 1. The sum of all probabilities for a single generator must add up to 1.*

Definition 2 (Probabilistic Decision Model (PDM))**.** PDM *in process simulation determines the direction at the XOR and OR gateways using predefined probabilities. For XOR gateways, exactly one pathway is chosen, while for OR gateways, one or more paths can be selected. Each possible pathway from a gateway has a probability assigned to it, where each probability is between 0 and 1, and the total of all probabilities for a gateway must equal 1. When a token reaches a decision point, the simulator acts as a dice roll to randomly select a pathway based on these probabilities.*

DAS integrates traditional BPMN elements with data attributes. This BPMN model outlines the control flow perspective, which includes process activities and their connections. In addition to this, data attributes address organisational, temporal, and data aspects. Although this paper primarily addresses the data components and decision-making processes, Definition 3 also covers the resource allocation and case-inter-arrival schemas, which are crucial for modelling the organisational and temporal aspects within the complete simulation framework.

Definition 3. *The Data-Aware BP Simulation Model (DAS) expands on typical BPMN elements $\langle E, A, G, F\rangle$ – incorporating events (E), activities (A), gateways (G) and flow arcs (F) – supplemented with distinct simulation parameters:*

- $RS = \langle \mathit{TRMap}, \mathit{ProcTimes}, \mathit{RAvail}\rangle$ *represents the Resource Allocation Schema, detailing how resources are assigned to tasks (TRMap), the duration needed for tasks (ProcTimes), and when resources are available (RAvail).*
- $AT_{AC} : \mathcal{P}(\mathbb{R}^+) \times$ *Intervals outlines the scheduling of case creation, combining* AT*, a function for the timing of new cases, with* AC*, the schedule defining when cases can start.*
- $D_g = \{g_1, g_2, \dots, g_m\}$ *is a set of global attributes. Each attribute* $g_i \in D_g$ *begins with the process and can be updated throughout its duration by any activity across all cases.*
- $D_c = \{c_1, c_2, \dots, c_k\}$ *comprises case-specific attributes, where each* $c_i \in D_c$ *starts with the case and retains its value within that case.*
- $D_e = \{e_1, e_2, \dots, e_l\}$ *consists of event attributes initialized at the start of each case, which may be altered by activities within that specific case.*
- $UR = \{ur_1, ur_2, \dots, ur_n\}$ *includes rules for updating attributes* ($ur_i : (D_g \cup D_c \cup D_e) \times \mathcal{S} \to \mathcal{S}$)*. Each rule applies logic and functions to modify attribute states within their domains.*
- $MUR : (A \cup \{\text{'Case Creation'}\}) \to UR$ *connects activities* $a \in A$ *and the case initiation process to their respective update rules, dictating how attributes are updated.*
- $HBC : G \times (\mathcal{S}_{D_g} \times \mathcal{S}_{D_c} \times \mathcal{S}_{D_e}) \to F$ *determines the path at decision gateways G based on the attribute states, employing logic and* PDM *to direct the flow through the process.*

In Definition 3, the attribute types D_g, D_e, and D_c outline how the data is accessible and manipulated within the simulation model. The case attributes (D_c) are local to each process instance and set once at the beginning of a process case, remaining unchanged. On the other hand, global (D_g) and event (D_e) attributes are dynamic, with global attributes affecting the entire process and event attributes being local to case.

The update rules (UR) define the procedures for initialising the case attributes and modifying the dynamic attributes. These rules adapt attribute values according to the process conditions using specific mathematical functions and algorithms. Essentially, update rules specify the conditions and methods for attribute changes, while mapping (MUR) dictates which process elements (activities or events) trigger these updates.

The term 'data state' in Definition 3 refers to a set of attribute-value pairs at any given moment in the process execution. This dynamic state can be altered by applying update rules, thus reflecting the current conditions within the process simulation.

Consider a patient treatment process from hospital admission to discharge to demonstrate DAS. Global attributes such as ***hospital capacity***, which tracks the availability of beds, play a crucial role in managing critical parts of the process, for example, handling new admissions when the hospital is full. Case attributes such as ***initial health status***, which captures the patient's health condition at admission, and ***assigned doctor***, indicating that the doctor responsible for the patient remains constant throughout treatment. In contrast, dynamic event attributes such as ***medication-administered*** and ***diagnostic test results*** document specific medical interventions over time, while ***patient evolution*** track changes in patient condition, such as moving from intensive care to a general ward.

Update rules modify attributes dynamically during the simulation based on ongoing events. For instance, ***hospital-capacity*** decreases with each new patient admitted and increases upon discharge. Changes in attributes such as the results of ***diagnostic tests*** or ***medication administered*** can be triggered in the evolution of the ***patient evolution***, influencing further treatment decisions.

These data attributes and update rules introduce a data perspective to simulation models, going beyond the traditional focus only on control flows, organisational structures, and timing. In data-aware models, the simulation engine generates event logs that include a wider range of process-specific data attributes, providing a more detailed view of the process. This approach improves detailed scenario analysis, offers insight into how data changes affect process outcomes, and aids in identifying potential data-related bottlenecks.

Despite the integration of data attributes and update rules, these elements alone do not modify other aspects of the process, such as control flow or organisational structures. To address this, Definition 3 introduces hybrid branching conditions (HBC). These conditions allow the control flow to change based on the current state of the data, potentially affecting the timing of the process. When the process reaches a decision gateway, the simulation engine uses conditions to decide the next arc based on the data accumulated up to that point. Here is a breakdown of how decision-making at gateways is handled in a data-aware simulation engine.

- **Two-Level Evaluation at Gateways:** Unlike PDM that relies only on probabilities, our data-aware model first evaluates conditions related to current attribute states to determine possible paths. To solve data inconsistencies (if any) it then uses a PDM to prevent simulation deadlocks.
- **Inclusive and Exclusive Gateways:** At XOR gateways, where only one path can be taken, the model uses a PMD to choose among multiple conditionally received flows. For example, 3 out of 5 outgoing flow conditions passed, and we chose only 1 out of 3 via PDM. For inclusive (OR split) gateways, which allow multiple parallel paths, the simulation proceeds with all flows where conditions evaluated 'true'.
- **Fallback to Probabilistic Evaluation:** If no conditions are evaluated as 'true' at a gateway, the simulation uses a PMD to continue the process, ensuring that decision-making is not blocked due to a lack of data-driven directions.
- **Preventing Infinite Loops:** To stop the simulation from getting stuck in infinite loops, if the same condition keeps being true, leading to repeated flow execution, the model changes to using PDM after a certain number of iterations to avoid endless loops.
- **Probabilistic Decisions Independent of Data:** The probabilities in secondary evaluations are independent of current data states, relying instead on typical frequencies of path selections.

The effectiveness of DAS depends on the ability of a simulation engine to utilise its data-aware components. Traditional simulation models often use fixed probabilities at decision points, which can lead to unrealistic and oversimplified scenarios [13]. For example, in a hospital process simulation, a traditional model might always send a fixed percentage of patients for MRI tests without considering each patient's specific health data. In contrast, a data-aware simulation engine would look at the unique medical details of each patient (such as the attribute ***patient evolution***), leading to results that more accurately reflect real healthcare decisions [14].

3.2 Data-Aware Simulation Model Discovery

3.2.1 Data Attributes Discovery

Attribute Classification We start by classifying the attributes based on the observed data attribute patterns in the raw event logs. Starting with the classification of attributes, our goal is to first identify and understand the simplest update mechanisms proceeding with dynamic data attributes that require a more complex discovery approach. The initial pattern we look for is attributes that maintain a consistent value throughout the entire simulation. These attributes are identified as **global attributes with fixed values**, as they are only initialised once and a constant value stream suggests there is no change over time. To identify **case attributes** we search for attributes that remain constant within a case, but set a new value at the beginning of a new case. Real-life data often contain noise, so we use a *confidence threshold* to determine how consistently an attribute must remain unchanged within cases to qualify as a case attribute. The classification of dynamic attributes, specifically **event and global attributes**, presents more complexity due to their changing nature. To handle this, we preprocess the event log into two versions, one assuming attributes are global (g_log) and the other treating them as event-specific (e_log). We then attempt to identify update mechanisms for both versions without prior classification, using machine learning models to determine which version - global or local - better fits the data. This approach helps us both classify the attribute type and discover its update mechanisms simultaneously.

Discovery of Update Mechanisms In the discovery of update mechanisms for simulation data attributes, we implement different methods based on the type and behaviour of each attribute.

1. **Fixed Attribute Discovery:** Global attributes with fixed value are straightforward to handle. These attributes do not require complex algorithms for their discovery. For each attribute categorized under this type, we directly extract and assign their constant value.

2. **Case Attributes:** Another straightforward approach is for case attributes due to their strict patterns and limited update mechanisms.
    - For attributes with continuous values, we use curve fitting for each value extracted from the case and discover potential distribution function that these values fit mostly.
    - For attributes with discrete values, we use frequency analysis to calculate probabilities of having each observed label.
3. **Dynamic Attributes:** The update mechanisms for dynamic attributes are more problematic due to their complexity and variety in update mechanisms. Although discrete values are still discovered with frequency analysis, continuous values are processed with several models simultaneously:
    - *Linear Regression:* This model is suitable for attributes that follow linear changes over time, effectively capturing any consistent increase or decrease in their values.
    - *Curve Fitting:* Still applicable for dynamic attributes to capture potential distribution functions.
    - *M5Prime (Regression Decision Tree):* For attributes that demonstrate complex, nonlinear changes, we use Regression Decision Tree. It constructs a decision tree of linear functions and is therefore able to deal with cases that cannot be discovered using simple linear regression and do not follow any distribution function.

    Once the models have proposed potential update rules, we assess their effectiveness using performance metrics (Earth's Moving Distance for continuous, and Kolmogorov-Smirnov Statistc for discrete values). For the actual classification and model selection, we need to do that process for each activity that modified an attribute and based on aggregated error model with the lowest value will be selected and classified accordingly.

Before implementing any update mechanisms, it is essential to select features for training and testing the model's performance. Since case attributes are independent of any values, we will use half of the dataset's values for training and the other half for testing, without extracting any features. In contrast, for global and event attributes, it is critical to enhance model performance by extracting additional features. This involves retrieving the previous value based on the type of log; for *g_log*, we arrange the entire sample by end time to ensure that the data are in sequential order, since attributes are determined at the end of an activity. If it is the first row or no previous value exists, we leave it blank. For *e_log*, the approach is similar but applied individually to each case, each case resetting the initial value to zero. Another vital feature is the difference, calculated by subtracting the previous value from the current one. This feature helps to eliminate activities where the attribute value remains unchanged. For case attributes, we solely use current values, but for global and event attributes, we incorporate previous values as features to predict current values. This approach yields linear functions for linear regression and regression decision trees with linear expressions that alter the attribute value. However, for Curve Fitting, which is also applied to case attributes, we continue to use only current values.

3.2.2 Branching Conditions Discovery

State Capture and Data Preparation During the discovery of branching conditions, we employ a replayer [15] to replicate the simulation behaviour and capture the state of the data attributes at the precise moment when a decision is made at the XOR and OR gateways. By simulating the decision points, the replayer allows us to record the values of attributes that influence the pathway selections at these gateways.

In addition, the replayer captures the pathways that are activated after each decision point. This data provides a direct link between the attribute states and the actual flow outcomes in the simulation. With this information, we construct a matrix, similar to that shown in Table 4, for each gateway, which outlines the attribute values at the moment of decision and shows if a particular pathway was activated or not.

Decision Tree Analysis From the datasets provided by the replayer, which include attribute values and results for all outgoing flows, we focus on analysing each flow individually. We identify conditions where a specific flow is activated, using decision tree analysis.

Table 4. Example of Gateway Dataframe

attr1	attr2	attr3	Flow1	Flow2	Flow3	Flow4	Flow5
A3	B5	C2	False	False	False	False	True
A2	B3	C1	False	False	False	True	False
A4	B2	C4	False	True	False	False	False
A2	B1	C5	False	False	True	False	False
A3	B5	C2	False	False	False	False	True
...	..	..	...	...	...	...	...
A4	B1	C4	False	True	False	False	False
A3	B1	C3	True	False	False	False	False
A2	B5	C1	False	False	False	True	False
A4	B3	C5	False	False	True	False	False
A2	B1	C5	False	False	True	False	False

Table 5. Decision Tree with all Outcomes

Conditions	**Outcome**
$[(i \leq 4000.5)]$	0
$[(i > 4000.5), (i \leq 4002.5)]$	1
$[(i > 4000.5), (i > 4002.5), (i \leq 4965.5), (g1 \leq 7445.25), (g1 \leq 7311.75)]$	0
$[(i > 4000.5), (i > 4002.5), (i \leq 4965.5), (g1 \leq 7445.25), (g1 > 7311.75)]$	0
$[(i > 4000.5), (i > 4002.5), (i \leq 4965.5), (g1 > 7445.25)]$	1
$[(i > 4000.5), (i > 4002.5), (i > 4965.5), (i \leq 4988.5)]$	0
$[(i > 4000.5), (i > 4002.5), (i > 4965.5), (i > 4988.5), (g1 \leq 7490.25)]$	0
$[(i > 4000.5), (i > 4002.5), (i > 4965.5), (i > 4988.5), (g1 > 7490.25)]$	0

We build a decision tree, for example, like in Table 5, where each path from the root to a leaf that indicates that a flow is activated (outcome = 1) helps us understand the conditions necessary to trigger that flow. We connect these conditions (conditions within one row) with the "AND" operator as we move deeper into the tree, refining the rule to be more precise. If there are multiple ways to activate the same flow, shown by different paths in the tree, we combine these conditions with the "OR" operator, in Table 5 it described as multiple rows. This means that the flow can start if any of these conditions are met. For example, as illustrated in Table 5, there are two favourable outcomes. In the final decision tree for this process, the two primary conditions are merged using the OR operator, whereas the components within each condition are connected through the AND operator.

We also simplify the rules by combining similar conditions afterward. For example, if we have conditions like $A > 50$, $A > 75$, and $A > 80$ all leading to the same result, we keep only the most restrictive condition, which is $A > 80$. If there is an upper limit like $A \leq 100$, we add that to make the final rule $A > 80 \wedge A \leq 100$. This makes our rules easier to read and more direct. Thus, following the example in Table 5, by filtering for true conditions and simplifying all conditions, we obtain the definitive decision tree presented in Table 6. Moreover, note that, in the final example, a single condition simultaneously involves two attributes (*i* and *g1*). Attributes of any type from the log can be freely combined.

Table 6. Final Conditions

Conditions	**Outcome**
$[(i > 4000.5), (i \leq 4002.5)]$	1
$[(i > 4002.5), (i \leq 4965.5), (g1 > 7445.25)]$	1

Edge Cases In the process of discovering branching conditions, we sometimes encounter edge cases that require special handling to maintain the integrity and functionality of our simulations.

1. **Always True Conditions:** Certain pathways may always activate, regardless of the input conditions. This can happen when conditions are inherently broad or obscured by a large number of interacting attributes. For instance, a scenario might involve a condition like $loan > 0$, which is always true due to the nature of the data when it's becomes impossible to detect which attribute is actually affecting the decision. In such cases, we assign a default condition that is always true.

2. **Absence of Detectable Conditions:** There are situations where decision tree analysis fails to identify any clear conditions that dictate the flow of a process, particularly in complex or noisy data environments. When no attributes significantly influence the decision at a gateway, formulating a specific branching condition becomes impossible for the discovery model. Whenever that happens, we utilise flow probabilities captured during the state analysis as a fallback mechanism in Prosimos using PDM. It ensures that the simulation does not stall or generate errors due to the absence of detectable conditions.

4 Implementation and Evaluation

In this Section, we carried out both testing and experimental evaluation phases. Testing ensures that the project's functions operate correctly and perform their intended tasks. However, to truly understand the impact of the DAS model on the simulation environment, we need to go beyond testing. This involves conducting experiments to assess the model directly. We used synthetic logs with known expected values to measure how closely our results match the original data, and run experiments on real-life logs to evaluate the model in an uncontrolled environment. Furthermore, we conducted comparisons between the DAS model and traditional models to assess the impact of the DAS model on KPIs.

4.1 Implementation

The simulation engine is implemented in an open source project available on GitHub[7]. Prosimos requires a BPMN model and a JSON configuration file as input parameters. The output includes an event log, a performance metrics file, and a simulation warning file. For detailed instructions on how to use its functionalities, refer to [4]. To access the simulation engine with the DAS model, consider using a particular branch [8].

Simod, like Prosimos, is available as an open-source project[9]. It functions by analysing event logs to discover and optimise BPMN models and to generate simulation scenarios compatible with Prosimos. The flexibility of Simod allows users to discover new BPMN models from logs or enhance existing models. For details on Simod's discovery algorithms, model optimisation capabilities, and integration with simulation tools, please refer to [6]. However, to access DAS model you will need to refer to another repository called **PIX Framework**[10] which aim is to aggregate functionalities of Prosimos, Simod and other projects in a single code base; thus our changes are stored there as a stand-alone functions that will be integrated in the project itself later. In addition, for Prosimos, there is functionality that supports the DAS model stored on a separate branch[11]

[7]https://github.com/AutomatedProcessImprovement/Prosimos

[8]https://github.com/AutomatedProcessImprovement/Prosimos/tree/parameters

[9]https://github.com/AutomatedProcessImprovement/Simod

[10]https://github.com/AutomatedProcessImprovement/pix-framework

[11]https://github.com/AutomatedProcessImprovement/pix-framework/tree/attribute_discovery

4.2 Testing

Testing procedures are carried out in Prosimos by performing unit tests on each functionality developed to support the DAS model. Given that the discovery functionality adopts a stochastic approach and cannot be directly tested, we have undertaken experiments to assess the outcomes of the discovery in subsequent sections.

4.2.1 Data Attributes Testing

The test of data attributes involves checking how attributes are represented and updated during the simulation process in the event log. We performed a series of tests to ensure that different attribute types follow their patterns and update rules are calculated correctly, considering the values of other attributes in the expressions.

Data Attribute Patterns in Event Logs We tested how different attributes—global, case, and event—are displayed in the event log. We have 15 test cases assessing all attribute types with different initialisation places within the log. We used a simple model with three sequential activities: A, B, and C. During testing, we will get a template of the simulation scenario and modify it, adding attribute generation in different areas of the model and creating different scenarios described in Table 7.

The acceptance criteria for these tests are as follows: for attributes that are set only once and remain constant, we directly evaluate the pattern observed after the simulation in the event log. For instance, for case attributes, we expect to see a pattern of A->A->A for each activity within the case, whereas for event attributes initialised in the middle of the case, the expected pattern is NULL->A->A. In situations where an attribute is meant to change, we examine the alteration in value following the activity that triggers the change.

All tests can be reviewed in detail on GitHub[12].

Testing Update Rules We performed unit tests on the attributes of events since it is the only place where the attributes can be changed by activities to validate the update mechanisms. We have several groups of tests described in Table 8 that focus on different aspects of expressions that cover arithmetic operations, comparison, string operation, and edge cases.

[12]`https://github.com/AutomatedProcessImprovement/Prosimos/blob/parameters/testing_scripts/test_attributes_interaction.py`

Table 7. Summary of Attribute Testing Configurations

Test Group	**Description**
Single Attribute Initialisation (6 tests)	Tests single discrete and continuous attributes initialised once globally, per event and per case to ensure consistent values throughout a simulation.
Multiple Attributes Creation (4 tests)	Validates that multiple attributes, both of the same type and mixed, are correctly passed to the event log without interference between their values.
Global Attribute Changes by Activity (2 tests)	Checks that global attributes are changed correctly by specific activities, either at the start of a case or by a designated event.
Multiple Global Attributes with Activity-Specific Changes (2 tests)	Ensures that multiple global attributes can be updated by specific activities without value confusion, maintaining attribute integrity throughout the process.
Mixed Attribute Updates (1 test)	Tests the interaction of global attributes with both case and event updates, verifying that values change at precise trigger points and reflect expected modifications.

The acceptance criteria are established as follows: a data structure is predefined with attribute names and their corresponding values. We compute a value through expressions and verify if it matches the expected result. For example, employing the *math* library, we evaluate the expression *"sqrt(attr1)"* where *attr1* is set to 49, expecting the outcome to be 7.

All tests can be reviewed in detail on GitHub[13].

4.2.2 Branching Conditions Testing

We designed a set of tests for the XOR and OR gateways to evaluate the gateway conditions. The test model in Figure 2 consists of a single gateway that leads to three possible outcomes. We use the same model but with an OR gateway to assess its functionality. This model allows us to simulate different scenarios of condition evaluation and easily detect all decisions because of the simplicity of the model.

Acceptance criteria for these tests include: In the setup, we specify the number of activities and activity names that should be executed and assess various scenarios.

[13]`https://github.com/AutomatedProcessImprovement/Prosimos/blob/parameters/testing_scripts/test_event_attributes.py`

Table 8. Summary of Update Rule Tests

Test Category	**Description**
Arithmetic Operations (8 tests)	Tests basic operations such as addition, subtraction, multiplication, division, modulus, exponentiation, true division, and floor division.
Comparison and Logical Operations (14 tests)	Includes tests for equality, inequality, greater than, less than, logical AND, OR, NOT, and comparisons involving zero or negative numbers.
String Operations (7 tests)	Focusses on string concatenation, multiplication, division by zero with strings, and logical operations with strings, including string comparisons.
Special Cases and Error Handling (5 tests)	Tests division by zero, handling non-existent attributes, logical not on an empty string, concatenation and multiplication of strings, and mixed-type addition.
Maths Functions and Misc. Tests (8 tests)	Covers mathematical functions, complex expressions, handling of special cases like exponential growth and log of negative, single-number output, and invalid expression handling.

For example, a single correct condition for an XOR gateway leads to just one activity with a specific name. On the other hand, when testing incorrect conditions, we still expect the activation of only one flow, but its name will be random due to the use of PDM, allowing any value from the predefined list to occur in the simulation.

We assess the functionality of gateway conditions using predefined attributes and conditions to determine the execution paths. In total, we have 12 tests, 6 for each type of gateway (XOR and OR), but considering their different behaviour and expected results in some cases:

1. All conditions false: Uses PDM to select paths.
2. All conditions true: Uses all true conditions to select paths for OR or selects one for XOR using PDM.
3. Multiple true conditions: Executes all true conditional flows for OR or selects one true condition utilizing PDM for XOR.
4. One true condition: Executes only the true conditional flow.

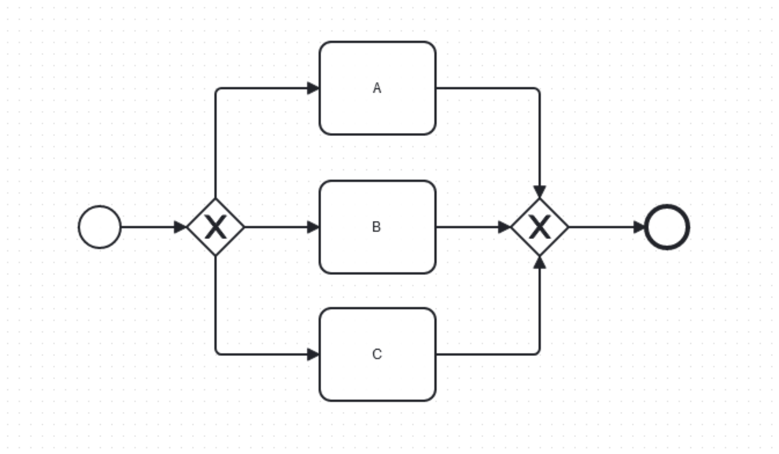


Figure 2. Branching conditions testing model

5. Only one condition available: Executes one conditional flow for both XOR and OR gateways to evaluate the hybrid strategy when not every path has conditions.

6. No conditions specified: Defaults to PDM path selection for both gateways.

All tests can be reviewed in detail on GitHub[14].

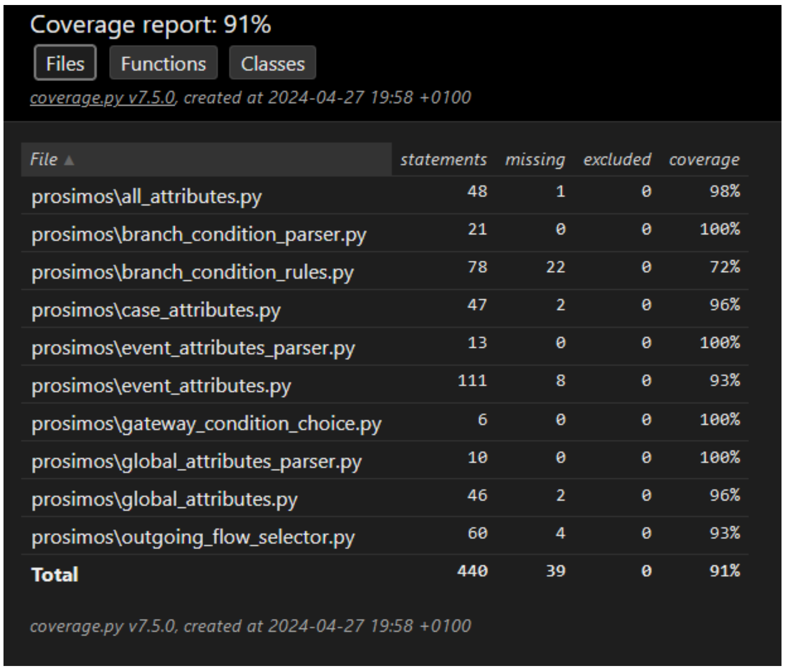
4.2.3 Code Coverage

After creating tests, it is essential to evaluate the extent of the code covered by these tests. This evaluation is referred to as *code coverage*, which helps to pinpoint the segments of our code that were not triggered during testing. Code coverage is a commonly used indicator for this objective, and we particularly track the proportion of code that has been activated at least once in the tests.

We used *pytest*[15] for this measurement, which allows testing to run and generate coverage reports concurrently. These reports can be presented in various formats, such as HTML, XML, or plain terminal output. We selected the HTML format for our coverage reports because it offers a user-friendly interface and better readability, making it easier to understand which parts of the code were covered by tests.

[14]https://github.com/AutomatedProcessImprovement/Prosimos/blob/parameters/testing_scripts/test_gateway_condition.py

[15]https://docs.pytest.org/en/stable/


Coverage report: 91%

coverage.py v7.5.0, created at 2024-04-27 19:58 +0100

File ▲	statements	missing	excluded	coverage
prosimos\all_attributes.py	48	1	0	98%
prosimos\branch_condition_parser.py	21	0	0	100%
prosimos\branch_condition_rules.py	78	22	0	72%
prosimos\case_attributes.py	47	2	0	96%
prosimos\event_attributes_parser.py	13	0	0	100%
prosimos\event_attributes.py	111	8	0	93%
prosimos\gateway_condition_choice.py	6	0	0	100%
prosimos\global_attributes_parser.py	10	0	0	100%
prosimos\global_attributes.py	46	2	0	96%
prosimos\outgoing_flow_selector.py	60	4	0	93%
Total	440	39	0	91%

coverage.py v7.5.0, created at 2024-04-27 19:58 +0100

Figure 3. Prosimos test coverage report

Figure 3 shows that we have achieved 91% statement coverage specifically in the newly added features but not throughout the project. Although the 91% coverage ensures that a substantial part of our new code has been tested, it does not directly measure how the features influence the simulation's performance. Additional experiments are required to measure the effects of the DAS model on the control flow and KPIs.

4.3 Evaluation

This section evaluates the DAS models implemented in Prosimos and Simod. After developing these models, which are designed to manage dynamic data and enable data-aware decision-making, we conducted a series of experimental evaluations aiming to determine the impact of the models on various aspects of the simulation, including control flow, cycle, and event times.

The evaluations for both Prosimos and Simod were conducted concurrently. This approach is necessary because assessing Simod's performance involves integrating and simulating the scenarios it discovers, thereby testing both the discovery and simulation engines. To evaluate the performance of the DAS models, we have developed a set of evaluation questions (EQs) to guide our analysis:

- **EQ1**: How accurately do we categorize the data attributes? This question focuses on determining the type of attributes based on their behaviour within the event log as a first basic step to evaluate the accuracy of the DAS model discovery.
- **EQ2**: How accurately can update rules be discovered? After categorizing the attributes, the next step is to identify and evaluate how closely the discovered update mechanisms match the intended behaviour in the DAS model.
- **EQ3**: What is the impact of the discovered DAS model on business process simulation? We evaluate the DAS model by assessing its impact on control flow and KPIs (cycle time and event time). The goal is to determine the accuracy of branching conditions and the advantages of using the DAS model over traditional models.

Cycle time is defined as the total time from the start to the completion of a process as measured within a simulation scenario. *Event time* refers to the duration of individual activities within the process.

4.3.1 Datasets

We used synthetic and real-life event logs to evaluate the performance of the DAS model. For synthetic logs, we prepare a simulation scenario and a BPMN model to create an original event log using Prosimos. The next step will be to use Simod to discover the simulation scenario with the traditional model (we do not rediscover the BPMN model and use the option to use the existing BPMN model). Then, utilising the code base from the PIX Framework, we discover data attributes and branching conditions and extend a copy of the traditional model with these results. After that, we use Prosimos again, passing discovered scenarios to get two more event logs, first generated with the traditional model and second with the DAS model. Then, we run a script to compute metrics by comparing the original log to the traditional and DAS models in pairs. The algorithm is the same for real-life logs, but we don't need to make an extra step in the beginning to generate the original log since it has already been provided.

Data Attributes For evaluating data attributes, we utilized the loan application BPMN model from Figure 4. The model includes 2 events(Start and End), 17 activities, 4 XOR split gateways, 2 AND split gateways, and 36 sequence flows. The diverse types of gateways and activities create many different pathways that a process might take, which is good for testing how attributes affect the process. This variety allows us to place attributes in different settings and see how they play out during the simulation.

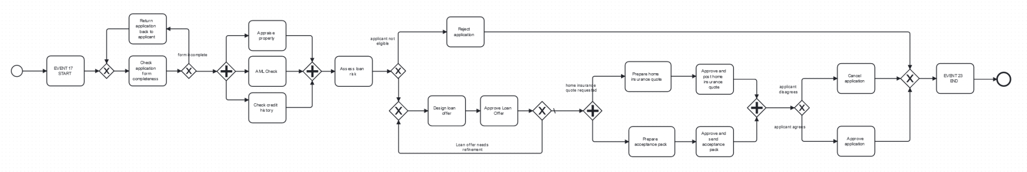


Figure 4. Loan application BPMN model.

For our experiments, we set up 82 data attributes: 10 for case attributes, 36 for global attributes, and 36 for event attributes. The small amount of tests for case attributes is because they have limited flexibility and do not support update rules. We test both old (generators) and new (update rules) features for global and event attributes. Here is a detailed breakdown of the tests for different attribute types:

1. **Case Attributes:** Each case attribute is identified by a test ID and a "c" label, which denotes a case attribute, followed by a short test name. We conducted 10 tests for case attributes using generators to create continuous values with different distributions—fixed, exponential, normal, and uniform (e.g., uniform distribution ranging from 0 to 100). We also tested discrete values with various labels and probabilities, such as having the probabilities "A":0.8 and "B":0.2. The primary goal with case attributes is to ensure that the generators operate correctly at the case creation level.

2. **Global and Event Attributes:** The naming convention for these attributes includes identifiers for the scope (global - g, event - e) and the frequency of update (once per case - s, multiple times per case - m), resulting in labels like se, sg, me, mg. We tested 18 different update mechanisms for each type, totalling 72 attributes for both global and event scopes. The tests for these attributes include the following.

   - **Generators:** Similar to the case attributes, we used generators for both global and event attributes to produce continuous values (using distributions like uniform from 1 to 100) and discrete values with probabilities (e.g., "A":0.8, "B": 0.2).

- **Update Rules:** These involve various expressions to simulate updates following linear functions (e.g., $1.05 \times x + 5$), periodic functions (e.g., $\cos(x)$), and complex expressions (e.g., $x + \log(x + 1) + 1$). We use linear functions to test our model on the capability to discover the easiest linear patterns as a baseline; then we increase the difficulty of the functions, making them periodic or adding more complex nonlinear aspects.

The loan application BPMN model and simulation scenario with all attributes used for the generation of synthetic logs is available in SharePoint [16].

Branching Conditions For branching conditions tests, we used a specially designed BPMN model, shown in Figure 5, which we set up specifically to handle decision-making scenarios. This model contains two events (Start and End), 19 activities, three XOR splits, and 39 flows. In that model, we clearly defined the spots where attributes have to be generated or changed right before the decision points and duplicated that pattern multiple times to add more weight within one case to calculate the KPIs more precisely. The model for assessing OR gateways is the same as in Figure 5 but with OR gateways instead.

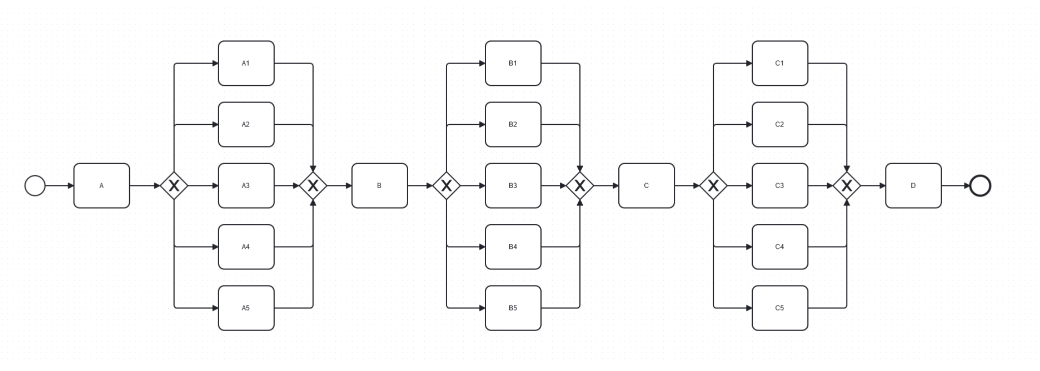


Figure 5. Branching conditions BPMN model.

In total, we conducted 59 tests (33 for XOR and 26 for OR gateways) testing

[16]https://tartuulikool-my.sharepoint.com/:f:/g/personal/murashko_ut_ee/EoamjGexw4tJg6qH3cOaQmMBFfyKtnJDlNNAy1QAAl809g?e=fu35Ka

conditions with different complexity or levels of noise. Detailed information on the test groups is described in Table 9

Some of the cases were created to evaluate the impact of noise on the evaluation of conditions. We tested two noise levels, 20% and 50%, for both the XOR and the OR gateways to analyse the results using a hybrid approach that combined branching conditions and probabilities. The percentage of noise is achieved by setting conditions within a specific range and omitting a certain percentage of the data range generated by the attribute. For example, to achieve 50% noise for an attribute with values from 0 to 100, we equally distribute the data ranges among arcs and then remove 50% from it. If the initial range was 0-20, we cut it to 0-10, and the next range will be 20-30. When the attribute generates a value of 10-20, we use PDM to simulate noise.

Our experimental setup also involved generating 5000 cases for each test to provide a comprehensive data set for analysis. We created a variety of simulation scenarios using scripts that automated the configuration for each test and another script to run all simulations using Prosimos to obtain all original event logs. Models and simulation scenario templates and scripts to generate configurations are available here in SharePoint[17]

[17] `https://tartuulikool-my.sharepoint.com/:f:/g/personal/murashko_ut_ee/EsZWKkC1jQ1Ii3kc6ivii00BvabzVI8YUh0O_C4MnRJraw?e=eBrbo4`

Table 9. Experiments for XOR and OR attributes

Test Category	Description
XOR Gateway: Discrete (3) + noise tests (6)	Assessing conditions based on discrete values. Tests include equally distributed values, non-equally distributed values, and scenarios where only one condition is evaluated as 'true. Noise tests examine the impact of random variations on these conditions.
XOR Gateway: Distributions (2) + noise tests (4)	Evaluating conditions based on normal and exponential distributions. Noise tests assess the robustness of these conditions under random variations.
XOR Gateway: Linear function (1) + noise tests (2)	Testing the impact of strict, deterministic conditions based on linear functions, and examining how noise affects the outcome.
XOR Gateway: Complex conditions (5) + noise tests (10)	Testing scenarios with multiple expressions that must be evaluated. These include combinations of AND and OR operators, such as attr1 Ā AND attr2 100. Noise tests analyze the impact of variability on these complex conditions.
OR Gateway: Discrete (5) + noise tests (6)	Similar to XOR discrete tests, but also evaluating cases where 1, 2, or 5 flows can be executed. Noise tests explore the effect of random variations in these scenarios.
OR Gateway: Distributions (6) + noise tests (4)	Testing normal and exponential distributions with conditions that select 1, 2, or all 5 flows. Noise tests examine how randomness impacts these selection conditions.
OR Gateway: Complex conditions (3) + noise tests (2)	Evaluating conditions using the AND operator with scenarios where 1, 2, or all 5 flows can be executed simultaneously. This involves intersecting branching conditions, e.g., flow 1 for range 0-30 and flow 2 for range 20-50 from a uniform distribution (0 to 100), creating a probability for simultaneous flow execution.

For each real-life log, we discovered both the BPMN model and simulation scenario using Simod with instructions available in the GitHub repository[18].

Sepsis Cases The Sepsis Cases dataset[19] is derived from a real-life event log documenting sepsis cases in a hospital. Sepsis, a critical condition usually triggered by an infection, represents a significant pathway through hospital processes, tracked by the hospital's Enterprise Resource Planning (ERP) system. The data set comprises approximately 1,000 cases, a total of around 15,000 events in 16 different activities. The log is rich with 39 different data attributes, including information on the group responsible for activities, test results, and checklist data. All events and attribute values have been anonymised to protect privacy, and although the timestamps of the events were randomised, the sequential order within each case remains unchanged to keep the logical flow.

The corresponding BPMN model consists of 16 activities corresponding to the various medical and administrative steps in managing sepsis within the hospital, 183 flows, 22 XOR and 21 AND split gateways. These gateways are essential for modelling decision-making processes in clinical pathways, especially for managing dynamic and time-sensitive conditions like sepsis. Hence, it provides a solid foundation for applying the DAS model, allowing us to explore how dynamic attributes and branching conditions affect the simulation of complex healthcare processes.

The Road Traffic Fine Management Process The Road Traffic Fine Management Process involves events related to road traffic fines, including notifications, payments, and appeals, managed by local police in an Italian city. This data set[20] allows us to explore changes in data attributes during the process of fine creation, notification tracking, and other related activities. It contains 11 data attributes, which are recorded only when they appear without further attribute tracing. This log provides a valuable case for examining data attribute discovery, pre-processing, and assessment of various types of data attributes. The model structure features 2 events (Start and End), 47 flows, 11 activities, and 18 gateways (8 XOR split gateways and 10 intermediate events). This complexity provides a good framework for testing branching conditions within the DAS model.

[18]https://github.com/AutomatedProcessImprovement/Simod

[19]https://data.4tu.nl/datasets/33632f3c-5c48-40cf-8d8f-2db57f5a6ce7

[20]https://data.4tu.nl/articles/dataset/Road_Traffic_Fine_Management_Process/12683249

4.3.2 Experimental Setup

EQ1-2 For evaluating the accuracy of attribute classification (**EQ1**) and the discovery of update mechanisms (**EQ2**), the used *loan application* model. The experiment involved constructing a simulation scenario based on this model and generating an event log with a predefined set of different attribute types. Also, the classification of global and event attributes is based on the discovery of the update mechanism. Hence, we must first evaluate the performance of the ML models. For that, we used the following metrics:

1. **Earth Mover's Distance (EMD)**: is used to evaluate continuous attributes by measuring the minimum work needed to transform one distribution into another. This "work" involves shifting distribution mass across a distance, capturing the differences between two distributions. It is used to check if the distribution of attributes generated by the simulation matches the expected distributions.

2. **Kolmogorov-Smirnov Statistic (KS)**: is used for discrete attributes by comparing the maximum difference between the cumulative distribution functions of two datasets. In dynamic attribute discovery, KS verifies if the behaviour of attributes, particularly their discrete distributions, matches the expected behaviours. KS is non-parametric, making it suitable for situations where data normality cannot be assumed.

After performing update mechanism discovery, we classify event and global attributes based on the best model performance, grouping attributes of the same type by the winning model. To address **EQ1**, we then gather all winning models and calculate the percentage of all attributes that have been successfully classified.

EQ3 To address **EQ3**, we compared pairs of simulated logs from the traditional and DAS models with the original logs. This analysis included examining the impact of the DAS model on control flow and KPIs using the following metrics:

1. **N-gram Distance [16]:** Measures the similarity between two logs by comparing their sequential patterns up to n-items long. For example, if the original log has the sequence "A -> B -> C" and the simulated log has "A -> B -> D," the N-gramme distance helps identify the differences in these sequences.

2. **Relative Event Distribution Distance [16]:** Quantifies the differences in event frequencies between the original and simulated logs. For example, if the original log has event "A" occurring 30 times and the simulated log has it 20 times, this metric highlights the frequency discrepancy.

3. **Cycle Time Distribution Distance [16]:** Compares the distribution of cycle times between the simulated and original logs. For example, if the original log shows that tasks typically take 2-4 days, but the simulated log shows 1-3 days, this metric measures the difference in these time distributions.

4.3.3 Experimental Results

Due to the extensive experimental data, we cannot present all metrics for each model and attribute type, and thus we depict them in summary tables. For detailed information, see the supplementary material [21].

Answering EQ1 Table 10 summarises the classification results for different types of attributes. The case attributes were accurately identified (100%) due to the stringent rules applied to this attribute type. For global and event attributes, 40 out of 44 continuous attributes were correctly classified, demonstrating high accuracy (approximately 90.9%). Discrete attributes showed lower accuracy, with 19 out of 23 correctly classified (approximately 82.6%). Misclassifications typically occur with attributes that exhibit stochastic distributions or those that are frequently modified within a case, reflecting the challenges in pattern recognition under such conditions. In general, the classification accuracy for all attribute types was 89.6

The main difficulties were with classifying global continuous attributes due to their stochastic distribution patterns. This was especially true for attributes modified by all activities within a case, making it hard to identify a consistent pattern. Discrete attributes faced similar challenges. In contrast, the event attributes showed the highest accuracy because their localised nature made it easier to generate consistent and abundant training data for model learning.

Answering EQ2 All results for **EQ2** are also available here[22] for continuous and discrete values.

[21] `https://tartuulikool-my.sharepoint.com/:x:/g/personal/murashko_ut_ee/ES5fk1Wg309DnIxLWmZ1zF4BCLUXzBmaavM594Ba7CIoZw?e=CZ6tII`

[22] `https://tartuulikool-my.sharepoint.com/:x:/g/personal/murashko_ut_ee/ES5fk1Wg309DnIxLWmZ1zF4BCLUXzBmaavM594Ba7CIoZw?e=CZ6tII`

Table 10. Simulation Data Attribute Type Classification Summary

Attribute Type	**Total**	**Correct**	**Incorrect**	**% Correct**
Global Continuous	22	18	4	81.8%
Event Continuous	22	22	0	100%
Case Continuous	3	3	0	100%
Global Discrete	12	11	1	91.7%
Event Discrete	11	8	3	72.7%
Case Discrete	7	7	0	100%
Total	**77**	**69**	**8**	**89.6%**

Discrete Attributes To address **EQ2**, we must take into account the following aspects:

1. **Significance of Update Frequency:** Attributes updated more frequently (groups "mg" and "me") showed smaller differences between the global and event log assumptions. The frequent updates made the distributions similar in both cases.

2. **Impact of Update Location:** Attributes updated at multiple points in the process, especially under global assumptions, showed more discrepancies due to the cumulative effects of each update. This was more noticeable compared to event log assumptions.

3. **KS Statistic Thresholds:** For evaluation, a lower KS value indicates a closer match between predicted and actual distributions. Generally, KS values below 0.1 to 0.2 are considered good, showing little divergence, while values approaching 0.3 or higher suggest a poor fit, indicating significant differences between the distributions.

The selected results of our experiments are summarised in Table 11, showing the KS statistic values for the global and event attribute assumptions in various configurations. Attributes prefixed with "sg" showed the highest discovery accuracy, as their shared, modified values throughout the simulation and the fact that they were only updated once made their patterns easier to detect. In contrast, the attributes "mg" and "me", which had frequent updates, faced significant challenges, leading to higher KS values and indicating poorer model performance and similarity in event logs. This underscores the difficulties in accurately classifying and discovering update mechanisms for discrete attributes, especially when updates are frequent and distributed between multiple activities.

Table 11. Average KS Values by Attribute Group

Group	Average KS Value
Global Multiple (mg)	0.229
Global Single (sg)	0.013
Event Multiple (me)	0.245
Event Single (se)	0.129
Case (c)	0.184

Continuous Attributes Our discovery tool supported three types of models with the hypothesis that they would effectively capture particular update rules. These models are as follows.

- **Linear Regression**: Suitable for attributes with linear growth or decline, modelling the relationship between an independent variable and a continuous dependent variable.
- **Curve Fitting**: Good fit for attributes following specific distributions, accurately modelling non-linear trends and periodic behaviours.
- **M5Prime**: A regression decision tree for complex, non-linear behaviours where linear and simple curve models don't fit well. It handles intricate relationships by splitting data into manageable subsets.

Table 12 shows a selection of results that highlight the performance of different models in discovering continuous attribute update rules. The focus is on different attribute type groups, highlighting the average EMD values for each group: "c", "sg", "se", "mg", and "me".

Table 12. Average EMD Values by Attribute Group

Group	Average EMD Value
Global Multiple (mg)	2.6686×10^{29}
Global Single (sg)	9.0784×10^{28}
Event Multiple (me)	21.9372
Event Single (se)	3.6048
Case (c)	3.5474

From an attribute-type perspective, event attributes performed the best across all function types due to their localised nature in a local scope, which allowed for consistent and accurate training data. Global attributes with complex growth patterns were the most challenging. The main issue was "context loss," where initial values are lost over time, leading to inaccuracies in long-term projections. Although the models could classify growth types correctly, they struggled to estimate initial values, resulting in high EMD values ‘. Excluding linear functions from the analysis and focussing on other experimental scenarios, it is evident from Table 13 that the average error reduces to levels comparable to those shown in Table 12. This suggests that significant errors arise predominantly in linear scenarios. A potential solution could involve integrating the initial values directly into the model, ensuring that both the training and testing phases keep the context of the initial value. The primary challenge lies in the training and testing methodology for time-series data, which requires a slightly modified approach compared to other data types.

Table 13. Average EMD Values for Global Attributes without Linear Functions

Group	**Average EMD Value**
Global Multiple (mg)	39.19
Global Single (sg)	6.41

By evaluating ML models' performance, Linear Regression excelled at handling linear functions, as it directly models linear relationships. This model achieved low EMD values, indicating that it closely matched predicted and actual values. Curve Fitting was most effective for attributes following specific distributions, accurately modelling expected distributions, and minimising EMD. M5Prime was the most adaptable for complex growth patterns, effectively managing intricate behaviours. However, it sometimes captured unnecessary data fluctuations, slightly reducing its overall accuracy compared to simpler models.

Answering EQ3 The objective of EQ3 is to assess the effectiveness of the DAS model versus the traditional (TR) probabilistic approach in various gateway configurations using specific performance metrics: n-gram distance (NG), event distribution (ED) and cycle time distribution (CT). This evaluation helps determine the precision and efficiency of the simulated process flows in capturing the intended dynamics of the modelled processes.

XOR Gateway Results

Table 14. Comparison of DAS and traditional models across performance metrics

Test Group	DAS-NG	TR-NG	DAS-ED	TR-ED	DAS-CT	TR-CT
XOR Pure Disc	0.0126	**0.0101**	**57.29**	80.926	**129.196**	205.593
XOR Noisy Disc	0.07	**0.0138**	176.79	**92.19**	392.55	**205.91**
XOR Pure Con	**0.0134**	0.0204	**81.5934**	131.6806	**181.7316**	286.7117
XOR Noisy Con	0.0632	**0.032**	261.7051	**148.1469**	419.712	**242.0827**
XOR Pure CC	0.0573	**0.0157**	165.509	**102.876**	348.871	**185.059**
XOR Noisy CC	0.0491	**0.0154**	115.417	**84.617**	221.545	**164.553**
OR Pure Disc	**0.0587**	0.2071	**721.406**	18514.181	**3046.362**	47814.659
OR Noisy Disc	**0.0786**	0.0933	**114.3098**	2049.58	**257.16**	4700.79
OR Pure Con	**0.0265**	0.2661	**983.5555**	17501.6663	**2907.6286**	53277.5724
OR Noisy Con	**0.0593**	0.0975	**193.3098**	3300.134	**332.4068**	5681.9588
OR Pure CC	**0.0221**	0.2821	**940.341**	15930.793	**2335.15**	42980.185
OR Noisy CC	**0.0248**	0.0931	**135.413**	2795.907	**264.215**	6175.817

- **N-Gram Distance (NG):**
  - *Pure Discrete:* The **NG** for the DAS model was 0.0126, only slightly higher than the TR model at 0.0101, indicating a strong adherence to expected control flows in noise-free conditions. This minimal difference underscores the effectiveness of clearly defined conditions in straightforward decision-making scenarios.
  - *Noisy Discrete:* With a significant rise to 0.07 in **NG** for the DAS model under noisy conditions, the increase compared to 0.0138 for the TR model highlights the DAS model's sensitivity to external disturbances. This suggests a potential area for improvement in robustness against environmental noise.
  - *Pure Continuous:* Here, the DAS model outperformed the TR model with an **NG** of 0.0134 versus 0.0204, reflecting precise condition handling in continuous data-driven environments.
  - *Noisy Continuous and Complex Conditions (CC):* Both settings under noisy conditions showed the DAS model struggling slightly with higher **NG** values (0.0632 and 0.0491) compared to TR model (0.032 and 0.0154), indicating struggles in complex scenario management.
- **Event Distribution (ED):**

  - The DAS model consistently managed to maintain closer event distributions in less noisy environments, as seen in the *Pure Discrete* and *Pure Continuous* setups, with **ED** values significantly lower than those of the TR model.
  - In noisy scenarios, the DAS model still managed to perform comparably or better than the TR model, indicating robustness in handling disturbances.

- **Cycle Time Distribution (CT):**
  - The **CT** were consistently better in the DAS model across *Pure Conditions*, suggesting that deterministic handling of conditions can significantly enhance process efficiency and predictability.
  - Noise introduced more variability, but the DAS model generally showed resilience by maintaining lower or comparable **CT** to the TR model.

OR Gateway Results

- **N-Gram Distance (NG):**
  - *Pure and Noisy Discrete:* The DAS model's performance was notably better in pure settings (NG of 0.0587) compared to noisy ones (**NG** of 0.0786), but both were significantly better than the TR model's performance, showing the critical role of conditions in managing multiple paths.
  - *Continuous and Complex Conditions (CC):* Even under complex conditions, the DAS model maintained lower **NG** values, underscoring its capability to effectively manage multiple, overlapping conditions.
- **Event Distribution (ED):**
  - In every case, the DAS model maintained more consistent and lower **ED** values than the TR model, particularly in the *Pure Complex Conditions* where they significantly influenced outcomes.
  - The OR gateway's ability to handle complex conditional logic was evident, with the DAS model demonstrating superior management of **ED** across all tests.
- **Cycle Time Distribution (CT):**

- The **CT** in OR gateways showed a distinct advantage for the DAS model, especially in managing complex conditions where multiple pathways are activated simultaneously.
- Despite the introduction of noise, the DAS model effectively minimized **CT** variances, providing more stable **CT** than the TR model.

The close performance of the DAS model XOR gateways to the TR model can often be attributed to the characteristics of XOR gateways. Since they only support one specific flow at a time, the aggregated error may not be significant enough to create a substantial difference even if the conditions do not work correctly. This means that deviations from the expected process flow are generally minor, as the XOR structure inherently limits the impact of incorrect or misfiring conditions.

However, OR gateways exhibit different behaviours when branching conditions are introduced. These conditions unlock previously unavailable capabilities within this type of gateway by enabling the execution of various combinations of flows. For example, with precise conditions set, it is possible to consistently activate three out of five possible flows in every process execution, but never more or less. This level of specificity in flow activation is not achievable with a probabilistic model, where there is always a chance that any number of paths from one to all could be executed at any given time, depending on the probability distribution assigned to each path.

This distinction highlights a significant advantage of incorporating conditions into OR gateways: They allow for precise control over complex process scenarios that probabilistic models simply cannot offer. Adjusting probabilities on the arcs in a probabilistic model does not grant the same level of deterministic control over which and how many paths are activated, illustrating a fundamental limitation in handling complex workflows with multiple concurrent paths.

Real-Life Logs The evaluation results can be accessed on SharePoint [23] featuring data attribute classification, discovery metrics, and comparisons with conventional models.

Sepsis Cases Overall, we discovered 27 attributes, including 1 case attribute, and the rest as global or event attributes, with 3 continuous and 23 discrete. Among the discrete attributes, 21 showed a distinct type difference with half the error

[23] https://tartuulikool-my.sharepoint.com/:x:/g/personal/murashko_ut_ee/ER2WWMfsE1dBiEu2kiilmSMBW4Tz1PZDgd0qfVk5EMBwyA?e=ABuW8q

rate compared to the other. However, in 2 instances, the KS value was nearly or exactly identical, possibly due to attributes being initialised at the start of each case, making it challenging or impossible to distinguish between these two attributes. However, most of the attributes maintained a low error rate of 0.125. Furthermore, based on this error rate, 21 attributes were identified as global, demonstrating the effectiveness of a global context in real data analysis, while 2 were identified as event attributes, indicating the presence of local case-specific data. This highlights the importance of examining every possible variation in the scope of the data. Notable variance was observed in the continuous attributes, particularly for *CRP*, where only the curve fitting proved effective, while the linear regression and regression decision trees failed, producing nearly identical poor outcomes.

Regarding the impact of DAS on the simulation results, the N-Gramme Distance analysis did not reveal significant differences compared to the traditional model, maintaining a similar control flow. However, the DAS model demonstrated a notable improvement in both the cycle time and event time distribution, reducing these metrics by approximately 30%. This indicates that the DAS model eliminated unnecessary flows that were contributing to longer cycle times and also promoted a more consistent control flow and activity initiation, which in turn led to a reduction in event time distribution. Furthermore, this improvement was consistently observed in all experimental trials, suggesting that the DAS model surpasses the traditional model in terms of KPIs.

The Road Traffic Fine Management Process We discovered 10 attributes without any case attributes. This data set showed the least accuracy in the observed distribution, with an average error of around 0.25. In addition, all attributes demonstrated problematic behaviours similar to those seen in the **Sepsis Cases** and were initialised at the beginning of the case, resulting in identical g_log and e_log metrics. In contrast, for continuous values, distinct attributes emerged as clear leaders in various scopes. The curve fitting identified both event and global attributes, achieving 30-40% lower errors in different scopes, while M5Prime excelled in handling non-linear cases, particularly with the (*amount*) attribute. Given that the current log contained no local attributes and numerous cases indicated that real data could adhere to both local and global scopes, as well as linear, distribution, and nonlinear patterns, our DAS model successfully identified these diverse types.

Unfortunately, the model did not discover any decisions based on the branching conditions, so we are limited to discussing only the data attribute perspective of the dataset without addressing how the branching conditions influence simulation performance. Essentially, since both models employ PDM, the results of the simulation performance remain unchanged.

5 Conclusion and Future Work

This thesis aimed to design the DAS model, enable its discovery from event logs and facilitate its simulation while maintaining the integrity of all components.

We achieved the design and development of a DAS model under **RG1**, which uses data attributes for decision-making, allowing dynamic adaptation to changes in data over time. Under **RG2**, we created algorithms to classify data attributes and discover their update mechanisms, along with identifying branching conditions for simulation scenarios from the event logs. For **RG3**, we configured the simulation engine to incorporate all components of the DAS model, allowing it to run simulations that include dynamic data-aware features. This setup maintains high integrity, ensuring that the models discovered by the discovery tool are ready for simulation after discovery.

Despite these improvements, the model faces certain limitations that offer opportunities for future research.

1. **Complex Attribute Behavior:** The discovery of global attributes behaving like time-series data presents challenges due to their complexity, as was shown in the evaluation. Future work could look into using more advanced machine learning algorithms that are better at dealing with such attributes, potentially making the model better at adjusting to continuous changes in the data.

2. **Attribute Update Flexibility:** Integrating generators within the update rules to enhance their flexibility will be another good enhancement. This integration could enable the use of generators for dynamic update mechanisms to utilize random values within expressions.

3. **Condition Discovery Precision:** The current model sometimes struggles to discover complex conditions accurately. Some conditions might be missing, while others contain redundant conditions with outliers. Improving the algorithm's capacity to identify accurate conditions can enhance the accuracy of the simulation, guaranteeing that all crucial decision paths are correctly modelled.

In conclusion, all research goals have been achieved, although there are areas where the current DAS model faces challenges. The proposed improvements will enhance the model's performance, helping it manage a wider variety of complex scenarios in business process simulations.

References

[1] Object Management Group. Business process model and notation (bpmn), version 2.0.2, January 2013.

[2] Orlenys López-Pintado, Iryna Halenok, and Marlon Dumas. Prosimos: Discovering and simulating business processes with differentiated resources. In Tiago Prince Sales, Henderik A. Proper, Giancarlo Guizzardi, Marco Montali, Fabrizio Maria Maggi, and Claudenir M. Fonseca, editors, *Enterprise Design, Operations, and Computing. EDOC 2022 Workshops*, pages 346–352, Cham, 2023. Springer International Publishing.

[3] Manuel Camargo, Marlon Dumas, and Oscar González-Rojas. Automated discovery of business process simulation models from event logs. *Decision Support Systems*, 134:113284, 2020.

[4] Iryna Halenok, Orlenys López-Pintado, and Marlon Dumas. Business process simulation with differentiated resources, 2023.

[5] Orlenys López-Pintado and Marlon Dumas. Discovery and simulation of business processes with probabilistic resource availability calendars. In *2023 5th International Conference on Process Mining (ICPM)*, pages 1–8, 2023.

[6] Ihar Suvorau, David Chapela de la Campa, and Marlon Dumas. Scaling out the discovery of business process simulation models from event logs, 2023.

[7] Jonas Berx, Orlenys López-Pintado, and Marlon Dumas. Optimisation of business processes with differentiated resources, 2023.

[8] Ken Schwaber and Jeff Sutherland. The definitive guide to scrum: The rules of the game, 2020.

[9] Marlon Dumas, Marcello La Rosa, Jan Mendling, and Hajo A. Reijers. *Fundamentals of Business Process Management*. 2 edition, 2018.

[10] Niels Martin, Benoît Depaire, and An Caris. The use of process mining in business process simulation model construction. *Business Information Systems Engineering*, 58:73–87, 02 2016.

[11] Sander P. F. Peters. *Analysis and Optimization of Resources in Business Processes*. PhD dissertation, Technische Universiteit Eindhoven, 2021.

[12] Suriadi Suriadi, Moe T. Wynn, Jingxin Xu, Wil M.P. van der Aalst, and Arthur H.M. ter Hofstede. Discovering work prioritisation patterns from event logs. *Decision Support Systems*, 100:77–92, 2017. Smart Business Process Management.

[13] Wil Aalst. Business process simulation revisited. volume 63, pages 1–14, 06 2010.

[14] Bedilia Estrada-Torres, Manuel Camargo, Marlon Dumas, Luciano García-Bañuelos, Ibrahim Mahdy Yousef, and Maksym Yerokhin. Discovering business process simulation models in the presence of multitasking and availability constraints. *Data Knowledge Engineering*, 134:101897, 05 2021.

[15] Orlenys López-Pintado and Marlon Dumas. Business process simulation with differentiated resources: Does it make a difference? In Claudio Di Ciccio, Remco Dijkman, Adela del Río Ortega, and Stefanie Rinderle-Ma, editors, *Business Process Management*, pages 361–378, Cham, 2022. Springer International Publishing.

[16] David Chapela-Campa, Ismail Benchekroun, Opher Baron, Marlon Dumas, Dmitry Krass, and Arik Senderovich. Can i trust my simulation model? measuring the quality of business process simulation models. In *Business Process Management Workshops - BPM 2023*, pages 20–37, Cham, 2023. Springer.

Appendix

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public