UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Ibrahim Mahdy Yousef

Mining Resource Availability for Data-driven Business Process Simulation

Master's Thesis (30 ECTS)

Supervisor: Marlon Dumas, Professor
Supervisor: Bedilia Estrada-Torres, PhD

Tartu 2021

Mining Resource Availability for Data-driven Business Process Simulation

Abstract:

Business process simulation (BPS) is a set of techniques to analyze a process model regarding identified performance metrics. BPS helps analysts decide whether to apply the process model to real-life production based on a set of statistics produced by a simulator. The accuracy of the output, accordingly, the simulation process's value withdrawn, is affected by the correctness of the business process model and the simulation parameters used as input. Therefore, data-driven simulation techniques are introduced to generate a process model from the process execution data recorded by the organization's information system to resemble reality. However, simulation models tend to be oversimplified due to some limitations of the existing simulation tools. One of the common limitations that need to be tackled is addressed towards resource availability and behavior. For example, it is assumed that resources are always available while, in reality, they have a work schedule, could have a part-time contract or are out of reach under certain conditions. In this respect, the existing business process simulators accept timetables to specify resource availability. On the one hand, providing a resource timetable to the simulator will increase the results' accuracy. On the other hand, a formal employee timetable most likely does not reflect the actual resource schedules. To this end, this research study presents a data-driven methodology that uses the execution log of the process under consideration to capture the reality of resource availability. The calendar discovery algorithm presented is integrated with the data-driven business process simulation tool, Simod [4]. As expected, the results show an increase in the precision of the discovered business process simulation model. The evaluation was carried out on four real-life logs as well as a synthetic data set.

Keywords:

Business Process Management, Business Process Simulation, Process Mining, Event logs, Resource Availability, Timetables.

CERCS:

P170- Computer science, Numerical Analysis, Systems, Control

Ressursside kättesaadavuse kaevandamine andmejuhitava äriprotsesside simuleerimise jaoks

Lühikokkuvõte:

Äriprotsesside simuleerimine on tehnikate kogum, mida kasutatakse protsessimudeli analüüsimiseks seoses identifitseeritud toimivusnäitajatega. See aitab analüütikutel simulaatori abil saadud statistiliste andmete põhjal otsustada, kas protsessimudelit tasub tegelikule tootmisele rakendada. Väljundi täpsust ja vastavalt simulatsiooniprotsessi pakutavat väärtust mõjutab sisendina kasutatud äriprotsessimudeli ja simulatsiooniparameetrite õigsus. Sellest tulenevalt kasutatakse tegeliku olukorraga sarnanemise tagamiseks andmejuhitavaid simulatsioonitehnikaid, mille abil luuakse organisatsiooni infosüsteemi registreeritud protsessi täideviimise andmete põhjal protsessimudel. Siiski kalduvad simulatsioonimudelid olemasolevate simulatsioonitööriistade teatavate piirangute tõttu olema ülelihtsustatud. Üks levinud piirangutest, millega tuleb tegeleda, on seotud ressursside kättesaadavuse ja käitumisega. Näiteks eeldatakse, et ressursid on alati kättesaadavad, samal ajal kui tegelikkuses on neil töögraafikud, neil võivad olla osalise tööajaga töölepingud või nad võivad olla teatavatel tingimustel kättesaamatud. Sellega seoses on olemasolevate äriprotsesside simulaatorite puhul võimalik ressursside kättesaadavuse määramiseks kasutada ajakavasid. Ühest küljest suurendab ressursside ajakava simulaatorile esitamine tulemuste täpsust. Teisest küljest ei peegelda ametlik töötajate ajakava suure tõenäosusega tegelikku ressursside ajakava. Sellega seoses esitatakse käesolevas töös andmejuhitav meetod, mis kasutab ressursside kättesaadavuse tegeliku olukorra kajastamiseks vaatlusaluse protsessi täideviimise logi. Esitatud kalendri avastamise algoritm on integreeritud andmejuhitava äriprotsesside simuleerimise tööriistaga Simod [4]. Ootuspäraselt näitavad tulemused avastatud äriprotsesside simulatsioonimudeli täpsuse kasvu. Hindamisel kasutati nelja reaalset logi ja lisaks sünteetilist andmehulka.

Võtmesõnad:

Äriprotsesside juhtimine, äriprotsesside simuleerimine, protsessikaeve, sündmuste logid, ressursside kättesaadavus, ajakavad.

CERCS: P170- Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction

Several companies and organizations have structured processes to run their business, and we can represent their processes as sets of events, activities, actors, and decisions performed to serve a specific goal. The art and science of observing how this work is carried out is called Business Process Management (BPM). BPM presents a vast set of tools, techniques, and methodologies, coming from various disciplines, including Industrial Engineering, Operations Management, Human Capital Management, and Information Systems Engineering, to support all stages of the business process [5]. Since the quality and efficiency of a corporation's services and processes depend on their process design and execution, BPM is essential for gaining customer satisfaction, outperform other competitors, and enhance internal processes inside the organization [5].

The research we are conveying is in the context of business process analysis. We address a technique for quantitative analysis of business processes, namely Business process simulation (BPS). BPS is considered a crucial technique during the analysis and redesign of the "to-be" process as it allows the generation of a large number of process instances in a relatively short time by recording each step of those executions. In this way, it is possible to try out new ideas to help organizations reach their goals and avoid failures and adverse outcomes. BPS is a well-known supported technique to analyze process models regarding performance metrics identified by cost, time, quality, and flexibility. Simulators generate a set of statistics reflecting the state of the process in terms of the process cycle time, and activities average waiting time, in addition to the utilization of the process resources [5].

Matin et al. [10] discuss that BPS models are obtained mainly through process analysts. They tend to revise the corporation's documentation, conduct interviews, and observe resources carrying out their work. Be that as it may, the actuality of the process under investigation could be biased as formal data may not reflect the actual process, interviews might lead to information conflicts, and employees' performance tends to hit a peek while being watched. Additionally, experts specify BPS parameters based on intuition, sampling, and manual curve fitting, which might lead to inaccurate models [17].

Consequently, it is vital to automate process models' discovery from the process execution data recorded by the organization information system (event logs), which is achieved by employing process mining (PM) techniques [19]. Adequate research has been put into that direction [10, 4, 12], since PM leads to discovering more accurate models that reflect the real process, including rare scenarios, in addition to fine-tuning the simulation parameters to fit the real event log.

Though many organizations already use automated business process simulation tools to analyze their processes, the simulation's effectiveness needs to be improved because BPS models tend to be oversimplified to accommodate some of the limitations of the existing simulators. Human resources' behavior in the process is one of the constraints that has not yet been entirely covered. For example, it is assumed that a resource reacts to the activity task once it arrives. The instant availability of a resource is not realistic since resources have a work schedule, tend to get involved in multiple processes, prioritize tasks, and work at a different speed [20].

In this thesis study, we tackle one limitation related to resource behavior, particularly the resource availability constraints. A resource availability constraint identifies the availability/unavailability of a resource or a group of resources during a specific period of time. A typical example of a resource availability constraints is that a resource is not available during the weekends or that the resource is not available before 9:00 or after 18:00. In other words, the resource is available only from Mondays to Fridays, everyday between 9:00 and 18:00.

Therefore, our main goal is to answer the following research questions:

RQ1. How to accurately identify the availability of resources based on business process event logs?

RQ2. Are data-driven simulation models that take into account resource availability more accurate than those that do not take into account resource availability, and to what extent?

The presented approach falls under process mining techniques, where event logs are processed to discover and extract the resource timetable. Our approach is based on the discovery of temporal patterns from time points by Yingjiu et al. [7].

We then use the discovered patterns to construct a resource availability timetable and inject it into the BPS model discovered by the simulation tool, Simod. The timetable's format is compatible with BIMP [8] simulator since Simod integrates BIMP to analyze the business process. Moreover, using our approach, we can discover the cases' calendar timetable by addressing the entire work events under the process, the resource pools' calendar where a timetable is generated for the process identified resource pools, and the cases' creation calendar timetable. We evaluated our technique using different data sets, including real-life and synthetic event logs, and the results' analysis demonstrates higher accuracy of process models that include the discovered calendar schedule compared to the process models that exclude it.

The rest of the document is structured as follows. Chapter 2 lays a background for further understandings of the document. The motivation for our research is encountered in Chapter 3, while Chapter 4 discusses the related work in the context of discovering resource availability. Chapter 5 explains the contribution of our work to obtain the resource availability calendar. The evaluation of the approach is reported in Chapter 6. Finally, a conclusion and direction for future work are drawn in Chapter 7.

2 Literature Review

This chapter reports the background notions needed to understand the remainder of the thesis. We define the main concepts of the business process, business process management, business process simulation, and the usage of data-driven in business process simulation, including a business process simulation tool used in our study.

2.1 Business Process

Every corporation providing a service or delivering a product performs a set of steps and activities which refer to their business process. Dumas et al. [5] explain that a business process encompasses a number of events and activities, while an event is an instantaneous incident that could lead to the execution of activities. In other words, an event does not have a time duration, in contrast to an activity, which refers to a unit of work presenting one or more tasks, each performed by a single process resource. For example, in an order-to-cash process, receiving an order confirmation is an event, while an employee (resource) preparing the order is an activity. A business process also includes decision points, which influence the process's execution workflow, physical objects (equipment or physical material.), information objects (e.g., electronic records), and actors participating in the process (a human, an organization, or a system).

The outcome of the process execution could be positive in case value is delivered to the process participants. However, if the desired output is not achieved or partially realized, this is considered a negative outcome. For example, in an order-to-cash process, a positive consequence is obvious for both parties when a customer places an order and receives the value for his/her money. On the other hand, if an issue leads to dissatisfaction to one or both parties, this is a negative result of the business process. Figure 1 displays the components and the outcome of a business process as Dumas et al. [5] illustrate.

Finally, enterprise organizations usually record the execution of their business process in a database system, which could be used to create a what is called *an event log* [22]. An event log keeps track of the process activity events where each

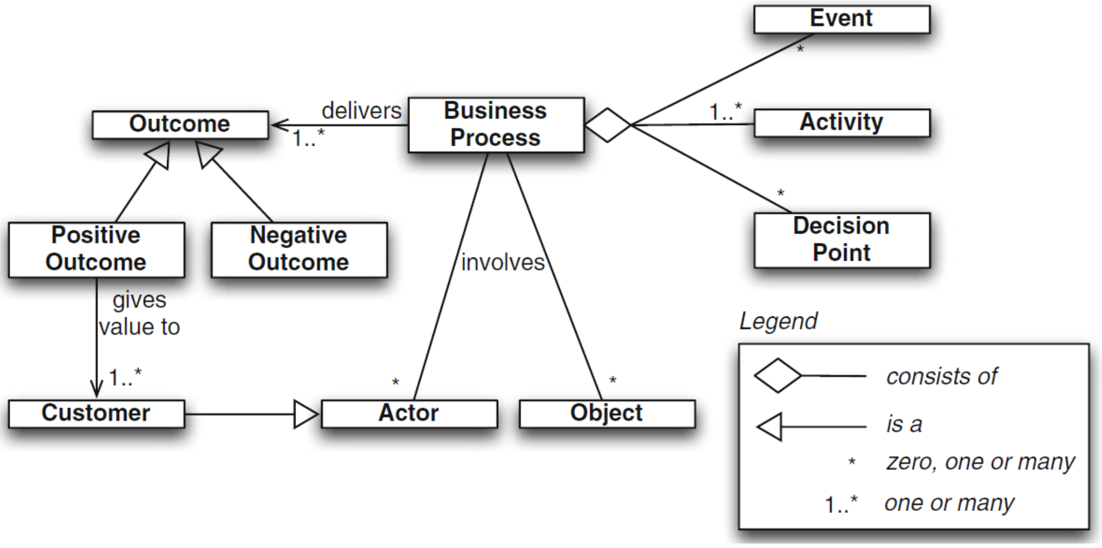


Figure 1. Business Process Components [5].

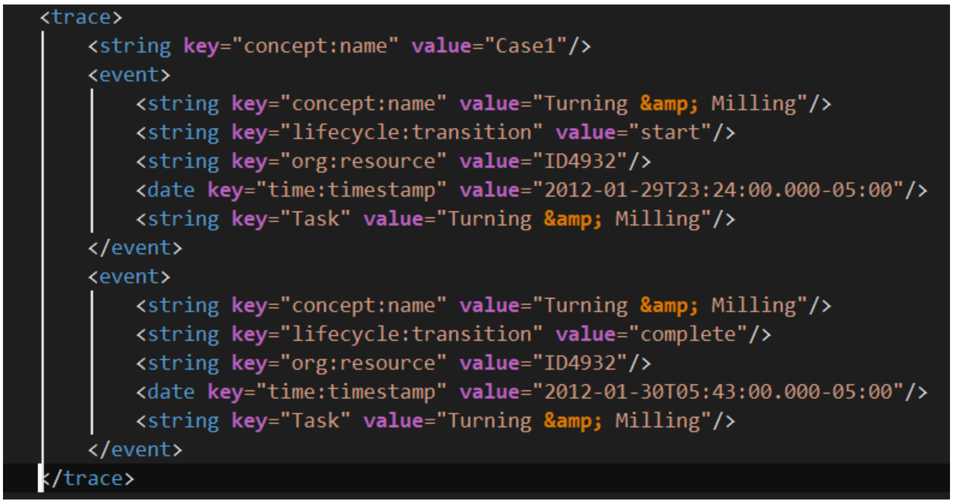
event captures the information of an activity task in the form of attributes. An event has to have at least the following attributes: i) an identifier (ID) ii) a case it is attached to (Case ID) iii) a timestamp to determine when the event took place. The most common format an event log is presented in is the eXtensible Event Stream (XES)[1]. An XES even log constructs of a set of *traces* where every trace has a number of events for the given case. Additionally, a trace could include attributes as well, in which case, they are shared among all of the trace's events (see Figure 2 for an excerpt of a trace).

2.2 Business Process Management

"Business Process Management is a body of methods, techniques, and tools to identify, discover, analyze, redesign, execute, and monitor business processes in order to optimize their performance" [5].

Business process management evolves around the business process, while all of the techniques and phases aim to ensure that the business process consistently drives positive outcomes and delivers maximum value to the organization and

[1] http://www.xes-standard.org.



```
<trace>
    <string key="concept:name" value="Case1"/>
    <event>
        <string key="concept:name" value="Turning &amp; Milling"/>
        <string key="lifecycle:transition" value="start"/>
        <string key="org:resource" value="ID4932"/>
        <date key="time:timestamp" value="2012-01-29T23:24:00.000-05:00"/>
        <string key="Task" value="Turning &amp; Milling"/>
    </event>
    <event>
        <string key="concept:name" value="Turning &amp; Milling"/>
        <string key="lifecycle:transition" value="complete"/>
        <string key="org:resource" value="ID4932"/>
        <date key="time:timestamp" value="2012-01-30T05:43:00.000-05:00"/>
        <string key="Task" value="Turning &amp; Milling"/>
    </event>
</trace>
```

Figure 2. Extract of a trace from an event log

its customers. A typical BPM process starts by identifying the business process, presenting it in an understandable format, then analyzing it to identify issues. Once issues are known, a process redesign is created and implemented to address them. Finally, the new process model is monitored to gain feedback. Dumas et al. [5] have defined that life cycle as follows (demonstrated in figure 3):

- **Process identification**: The goal of Process identification is to have as a result a new or updated process architecture with an overall view of the organization's processes and relationships. The output of process architecture is then used to select which processes to manage.

- **Process discovery**: A Business process is represented using one or more of the standardized format in the form of flowchart diagrams such as Business Process Modeling Notation (BPMN) [1], Activity Diagrams or any other flow chart format. The process here is documented in its current state that is called the "as-is" process.

- **Process analysis**: The goal of process analysis is to identify the issues and document them. Issues are measured based on performance measures that are related to cost, time, quality and flexibility. Prioritizing the issues is part of this phase as well to target the most important ones. Prioritization is based on the potential impact and the estimated effort required to solve them.

- **Process redesign**: In this phase, we answer the question: *What changes need to be done to the process to address the issues identified in the Process analysis phase?* Redesign options are considered based on the process analysis results to generate a “to-be” process model. It would be ideal if the new model is tested first before being applied to the real world to avoid any negative outcomes and reach the best results, which could be done using Business Process Simulation tools.

- **Process implementation**: The purpose of process implementation is to make the “to-be” process applicable to the real world. This might require the organization to adopt the way the “to-be” process is executed. In addition, the process itself could be changed by being automated, for instance - which would involve the IT department in the company - to support the enhanced process.

- **Process Monitoring**: *How well the “to-be” process is doing?* We monitor, inspect and analyze the redesigned process to gain feedback with respect to the expected behaviors. If issues appear then we repeat the BPM cycle to address them.

Upon redesigning the business process, it is required to implement a safe-test before integrating it into the organization. This is to ensure keeping any negative outcome out of the way and acknowledge positive results. Business Process Simulation is introduced to achieve that purpose.

2.3 Business Process Simulation

Business process simulation (BPS) is part of the process analysis phase; it allows the organization to emulate the actual process to check the results of the proposed changes by using a computer system [10].

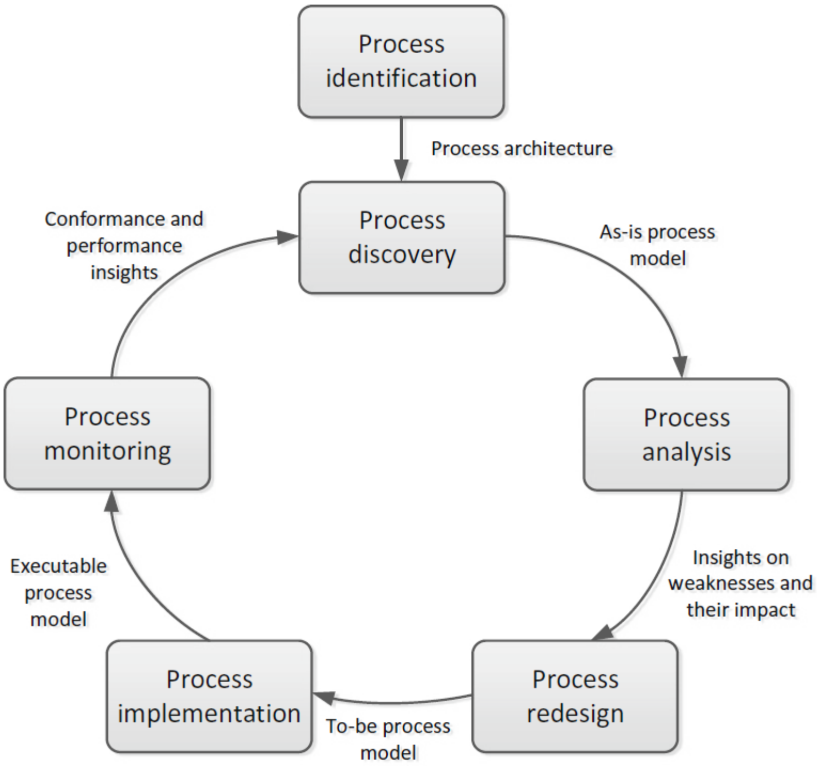


Figure 3. The Business Process Management lifecycle [5].

As explained by in [5], a simulation works as follows. Once a task is ready for execution, a so-called work item is created, and it is assigned to an available suitable resource. If there are none available to carry it out, it is put into waiting mode. After the work item is assigned to a resource, its duration is specified by the simulator based on a probability distribution defined as a parameter of the simulation model. The work item's status is set to "being executed" and then to "completed" once it is finished, and the resource becomes available again. As mentioned before, the target of a simulation process is to compute statistics about tasks. Therefore, the simulator keeps track of three important time attributes; the time when the task was ready for execution, the start time, and the task's end time. This data is used to compute the desired information, such as the average waiting time and resource utilization. These metrics are used to determine factors as the bottlenecks in the process and the percentage of time a resource is busy.

For a simulator to work, we provide a business process model and the following information data for each task, as specified in [5]:

- **Probability distribution for task duration**: It could be *fixed* if the processing time of the task is always the same, such as in the case of automated tasks, or it could be *exponential* if the processing time is around a given mean value. Finally, it could follow *normal distribution* in which the tasks' processing time is around a given average with a deviation value that allows the processing time to either be above or below the mean given value. The mean value and standard deviation are specified by the analyst as parameters of the simulator.

- **Resource pool**: Participants who are responsible for carrying out the task. Also, how many participants are available for each identified resource pool is provided.

- **Branching probability**: Branching probability is used for each decision gateway to determine the probability of the flow coming out of the gateway.

- **The mean inter-arrival time**: The mean inter-arrival time defines the time interval between the starting time of a process instance and the starting time of the following one. It follows a probability distribution function specified as an input of the simulator along with the value of the inter-arrival time.

- **Start date and time of the simulation**.

- **End date and time of simulation or duration of the simulation**: When the duration is specified, it would be added to the start time to derive the end time and date. When the end time is reached, the simulator stops producing instances.

There are several simulation tools available in the market, some of the popular ones include Appian, ARIS, Signavio Process Manager, and BIMP[2], and they all provide simulation capabilities with close features. In this research, we use BIMP [8]

[2] Appian, ARIS, Signavio Process Manager, BIMP.

simulator, not only due to its simplicity and suitability for academic research but also because BIMP provides the main functionality a commercial tool offers. It is worth noting that BIMP is not integrated directly into the calendar discovery algorithm. Instead, Simod [4], which automates the discovery of BPS models from event logs, uses the calendar discovery algorithm to retrieve the resource availability timetable and inject it into the discovered business process before feeding it to BIMP. BIMP receives the BPMN structure of the process alongside the simulation parameters in the form of XML data or by providing them through the web interface[3]. The simulator then produces an analytical output that takes different forms based on the tool, usually presented as a log to be analyzed or a set of diagrams with process statistics. In the case of BIMP, it generates statistics about the process cycle times, waiting times, resource utilization, and costs in the form of histograms (see Figure 4).

As mentioned in Chapter 1, the accuracy of the simulation model, and the simulation parameters plays a crucial factor in the accuracy of the business process simulation results. Consequently, the science of data-driven is introduced into the simulation process to ensure as precise results as possible.

2.4 Data-driven Process Simulation

Data-driven process simulation is a set of techniques used to create a BPS model from the data recorded by the organization's information system that reflects the business process details. Since organizations usually record their process execution information in the form of event logs, we make use of data-driven process simulation to extract insights from such process data to construct BPS models. In general, techniques for gaining knowledge from event logs belong to the process mining (PM) field, while using process mining for simulation fields also relates to several use cases in the field of BPM [21]. Therefore, PM techniques have been consolidated into the business process simulation field [9, 10, 16]. Martin et al. [10] present the state of the art of using PM in BPS by introducing four main BPS model building blocks:

[3] BIMP web interface is available at http://bimp.cs.ut.ee

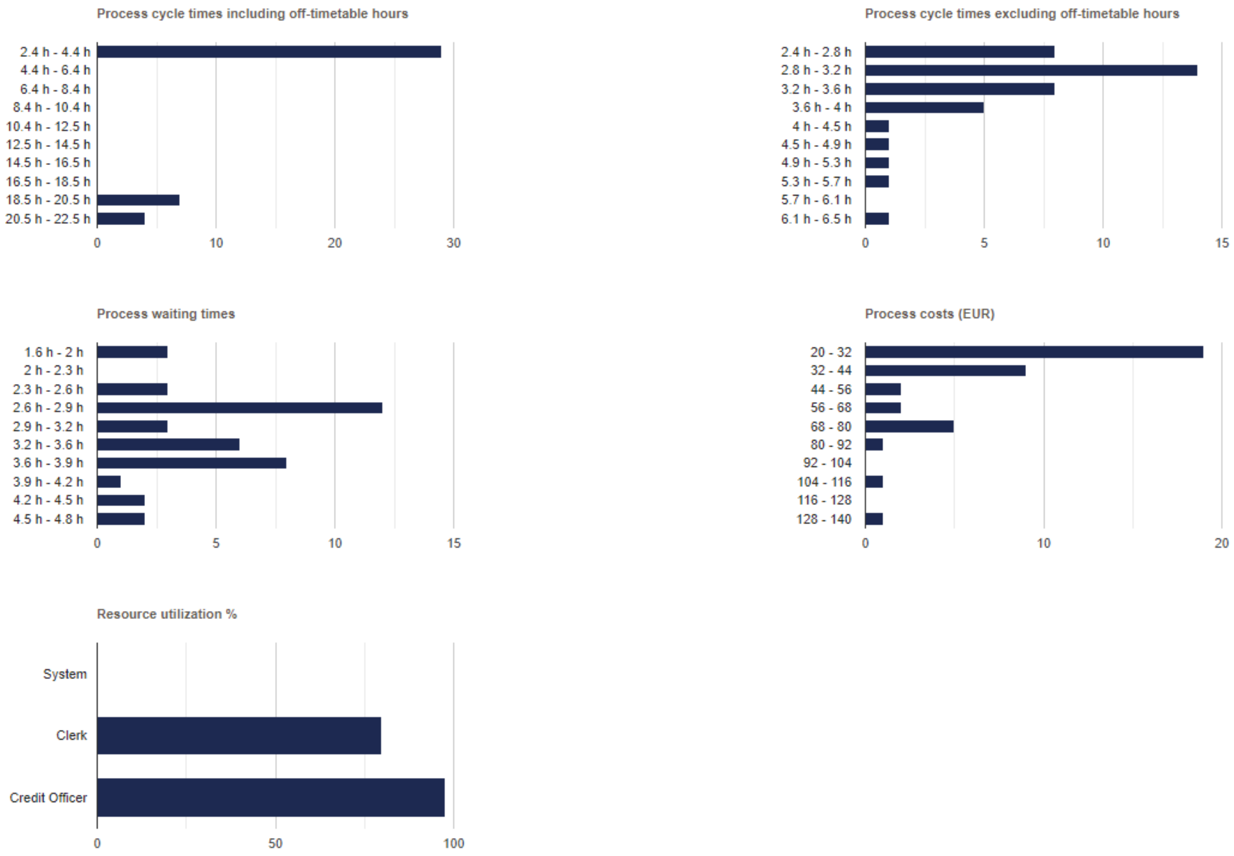


Figure 4. Histograms created by BIMP simulator for a credit card application process

- **Entities**: Model activities are executed on objects which flow through the system. These objects are called entities, and each of them has attributes, type, and arrival rate. Entity attributes are modeled through event log attributes. Entities with matched or similar attribute values are grouped by a clustering algorithm to define an entity type which makes understanding the logic behind the model easier.

- **Activities**: Since events in the log are saved in the form of activities, determining which of them should be included in the BPS is vital for the accuracy of the model. Therefore, activity mining techniques are required to identify the appropriate set of activities. In addition, they are used to map the processes information to the selected activities. For each activity, activity duration, resource requirement, queue discipline, and activity interruptions are determined from the event log.

- **Control Flow**: Control flow specifies the route an entity takes through the process. The followed path shows the activity sequentiality, choice, and concurrency. Gateways probability is determined since the flow choice made on a gateway determines the routing logic. Gateway routing logic is represented by a specific probability, business rules, or a combination of both.
- **Resources**: Resources execute the activities in the simulation model. Resource role simplifies the simulation model by grouping resources carrying out similar activities together. Activities are assigned to a resource role rather than to a single resource. A resource schedule could be presented to identify the availability of a resource. The resource availability follows a working day timetable. Moreover, batch processing is another feature that shows that entities are accumulated then processed either concurrently, sequentially, or simultaneously.

Simod [4] is a data-driven process simulation model based tool, to automatically generate simulation models from event logs. The tool first discovers a process model from the event log using automated process discovery techniques. Simod then extracts the simulation parameters using trace alignment and replay techniques. In addition, the tool makes use of hyper-parameter optimization to enhance the accuracy of the generated simulation model through the search for the best configuration of the simulation model parameters. The BIMP simulator can then process the generated BPS model.

Simod has three stages, pre-processing, processing, and post-processing. In the *pre-processing stage*, Simod discovers a BPMN model from the event log. This step is crucial since the accuracy of the simulation model is highly affected by the accuracy of the business model discovered. Simod makes use of the SplitMiner algorithm [2] to produce the BPMN model. SplitMiner discovers the model based on the predefined input parameters epsilon and eta. Epsilon is the parallelism threshold that determines the number of concurrent relations between events to be captured. Eta is the percentile for frequency threshold that filters the incoming and outgoing edges of each node to keep only the ones satisfying eta value. After the model has been discovered, Simod measures the conformance between the process model and the event log to evaluate how much the model fits with the event log. This is done using conformance checking fitness measure [13]. Simod takes care of non-conformant traces by removing, replacing, or repairing them according to the approach presented in [15]. Removal means deleting non-conformant traces from the log resulting in a reduced log containing conformant traces. Replacement is done by changing the non-conformant traces into the most similar conformant ones to preserve the number of traces. Repairing is done by aligning non-conformant traces in the event log using the technique in [14].

In the *processing stage*, simulation parameters are extracted and added to the BPMN to construct the BPS model. The parameters extracted are the ones required by the BIMP simulator. First, Simod replays the repaired event log against the obtained BPMN and calculates the frequency of traversal of the conditional branches, which is used later to compute branching probabilities. Simod also discovers the inter-arrival time of tasks. Activity processing time and their probability distribution function are extracted as well. Finally, the tool finds out the process resource pools using the algorithm presented in [18]. Once these parameters are calculated, Simod merges them with the BPMN model to construct the BPS model. The created BPS model is then fed into the BIMP simulator.

The *post-processing* stage aims to measure the accuracy of the generated simulation model compared to the original event log used as input in the pre-processing stage. The result is provided to the user to judge whether or not to use this model.

It is worth noting that a change in the parameter inputs in the pre-processing stage, epsilon, and eta, would result in a different structure of the process model (which would affect the simulation model accuracy in respect to the original event log). Therefore, Simod provides a hyper-parameter optimizer using the Tree-structured Parzen Estimator (TPE) [3]. This tool explores the search space to find a suitable combination of parameters that yield the most accurate simulation model.

The resource availability calendar presented in this research is integrated into Simod. Simod calls the calendar discovery algorithm to retrieve the resource timetable and append it into the process model. The discovered process model, as mentioned, is then passed to BIMP simulator.

In the next section, we highlight the limitation of taking resource availability constraints into account. We address how resource availability is a crucial element that will increase the business process model's precision and, consequently, the simulation results' usefulness for the corporation's business process.

3 Motivation

Business process simulation models tend to oversimplify things due to limitations in the existing simulation tools. One constraint in the process simulation approaches lies in the behavior of resources, since it is challenging to simulate scenarios that include cases where people are involved in multiple processes; resources that do not work at a constant speed; employees who try to group similar task in batches; resources that assign priorities during the selection of tasks to be executed; resources that work on weekdays only or have a part-time job [20].

As explained in Section 2.3, for a task to be executed, it is assigned to a suitable, available resource. The determination of the availability could be based on a timetable provided to the simulation tool. Otherwise, the simulator assumes the instant availability of the resource. In other words, the resource is always ready to carry out tasks nonstop (as long as the resource is not busy performing another task at the moment). Nevertheless, this assumption does not hold in reality because resources work mostly according to a schedule - in a typical organization, from 9:00 to 17:00 excluding weekends - not to mention lunch and coffee breaks, getting interrupted, or even flexible working hours.

Moreover, the inconsideration of availability constraints and the expectation of 24/7 availability leads to inaccurate cycle time distribution and a simulation model that does not reflect the reality. Consider the following scenario, the autumn semester's application period in a university is from January until the end of February. Therefore, a part-time employee is hired to assist applicants through the application process and reply to their questions during this period. The employee is present in the office five hours a day (from 9:00 to 14:00, from Monday to Friday). It is noted that a considerable number of emails arrive during the evening hours and over the weekend. Consequently, emails arriving over the weekend and evenings have to wait until the employee is available again, leading to a longer cycle time. Furthermore, the employee's workload is heavier on Mondays, handling the ones that arrived over the weekend, which means that requests arriving on Mondays have a slower cycle time than the ones reaching out in the middle of the week. Taking this scenario into account, the day-and-night availability of resources over the process's period assumed by the simulator does not match the reality of the seasonal availability with part-time hours, which also leads to an inaccurate distribution of cycle time.

Accordingly, we hypothesize that taking resource availability constraints into account will improve the simulation model accuracy and, therefore, the effectiveness of the conclusion drawn from the simulation results. This research study will discover a calendar to define resource availability under the current process. The proposition is data-driven and fully automated through integration with Simod. The impact is then examined to verify a positive influence on the generated simulation models.

Finally, before stating our contribution in this matter, we walk through the related work that has been done with respect to retrieving resource availability in the next section.

4 Related Work

The creation of business process simulation models that align with reality requires taking resource availability constraints into considerations, as explained before in chapter 3. In most cases, resource availability specifications are drawn from formal timetables or provided by domain analysts. However, formal timetables reflect the actual process only in environments with a fixed arrangement such as airports or hospitals. Also, relying on expert's intuition or research could lead to biased information [20]. Therefore, data-driven approaches to derive resource availability from the process execution logs such as the one presented in this thesis are needed.

Research studies related to discovering resource availability are considerably low. The only data-driven method that addresses this topic is presented by Martin et al. [11], where a calendar timetable is retrieved to show the periods during a working day when a resource is available and when he/she is idle or unavailable. The respected approach follows two steps to acquire the availability of resources. In the first step, what is called *Daily availability records* is retrieved that conveys for each resource the availability and unavailability times during each working day over a time period. In other words, a given resource is determined to either performing a task, ideal (available but not performing a job item) or unavailable to process a waiting piece of work. In the second step, the availability calendar for each resource throughout the time period is constructed by performing a sampling process on the daily availability records. The classification is done through direct sampling, which uses random sampling techniques, or cluster-based sampling using a clustering methodology instead of random selection.

Though Martin et al. [11] imply that their approach could be used in the context of business process simulation, it is not evaluated whether and to what value the correctness of the generated BPS model's will be improved. One of their approach's limitations is that the extraction of resources' calendar is limited to retrieving the availability for each singular resource, while most simulation tools operate on the resource pool level. On the other hand, we can discover the calendar for each individual resource or the entire resource pool. Also, we can discover the case calendar if no resource pool is provided. A second constraint of the method by Martin et al. [11] is the level of exploration. In other words, they answer the question: *what are the working hours of a resource during a specific period?* On the contrary, the approach presented in this research allows the definition of one or multiple time granularities, which gives flexibility to the level of exploration. That means the calendar is traversed based on a predefined time granularity such as year, month, weeks, days, hours, or minutes. For example, it could be retrieved that an employee worked at the organization from January until May given the granularity level is “Month”, avoiding unnecessary details about the working days and hours. One more limitation is that the accuracy of the generated calendar by their approach is limited by the accuracy of the sampling techniques used. In comparison, we allow the tuning of the accuracy for the discovered calendar by constraints of *support* and *confidence* values, which tightens or widens the explored space of timepoints. Finally, in addition to obtaining the resource availability calendar, our technique can discover the calendar related to the creation of cases, unlike the method provided in [11].

In the proceeding chapter, we detail our contribution to address the resource availability constraints.

5 Contribution

This study aims to discover the resource availability timetable from the process's event log. We use as an initial start the proposal by Yingjiu et al. [7], which discovers temporal patterns from a set of time points. We then use the discovered patterns to construct a resource availability timetable. In the following sections, we define the theoretical background of the calendar discovery and then describe the algorithm implemented to reach our goal.

5.1 Fundamental of Calendar Discovery

Our approach for retrieving the resource calendar is based on discovering calendar patterns from a set of time points presented in [7] by Yingjiu et al. They explain that events repeat over time according to a temporal pattern. For example, a company carries our maintenance operation for its servers every 10_{th} of the month. Such regularity could then be captured by a time granularity or multiple granularities, in this example, month and day time granularities. The following definitions are adopted from [7] to explain the fundamentals behind the discovery of resource timetable from an event log.

Definition 1 (Granularity Expression - GE). *A granularity expression is a set of time granules that do not overlap. Granularity expression takes the form of* $(g_n, g_{n-1}, g_{n-2}, .., g_2, g_1)$ *where each field is a granule subset of a time domain* [4]*.*

Each granule of a valid granularity expression in g_n has to be a union of some granules in g_{n-1}. In other words, granule g_2 has to cover g_1 in the way that the time domain of g_1 is a subset of the time domain of g_2. For example, (*Day, hour*) is accepted since a day consists of a set of identified hours, and an hour is covered by the day it falls in. However, (*Week, Hours*) is unjustifiable since a week does not contain a set of unique hours.

A granularity expression is identified as input to the algorithm by the user. Figure 5 shows examples of granularity expressions. For instance, (*Year, Month,*

[4] Examples of time granules are *"Year", "Month", "Week"*.

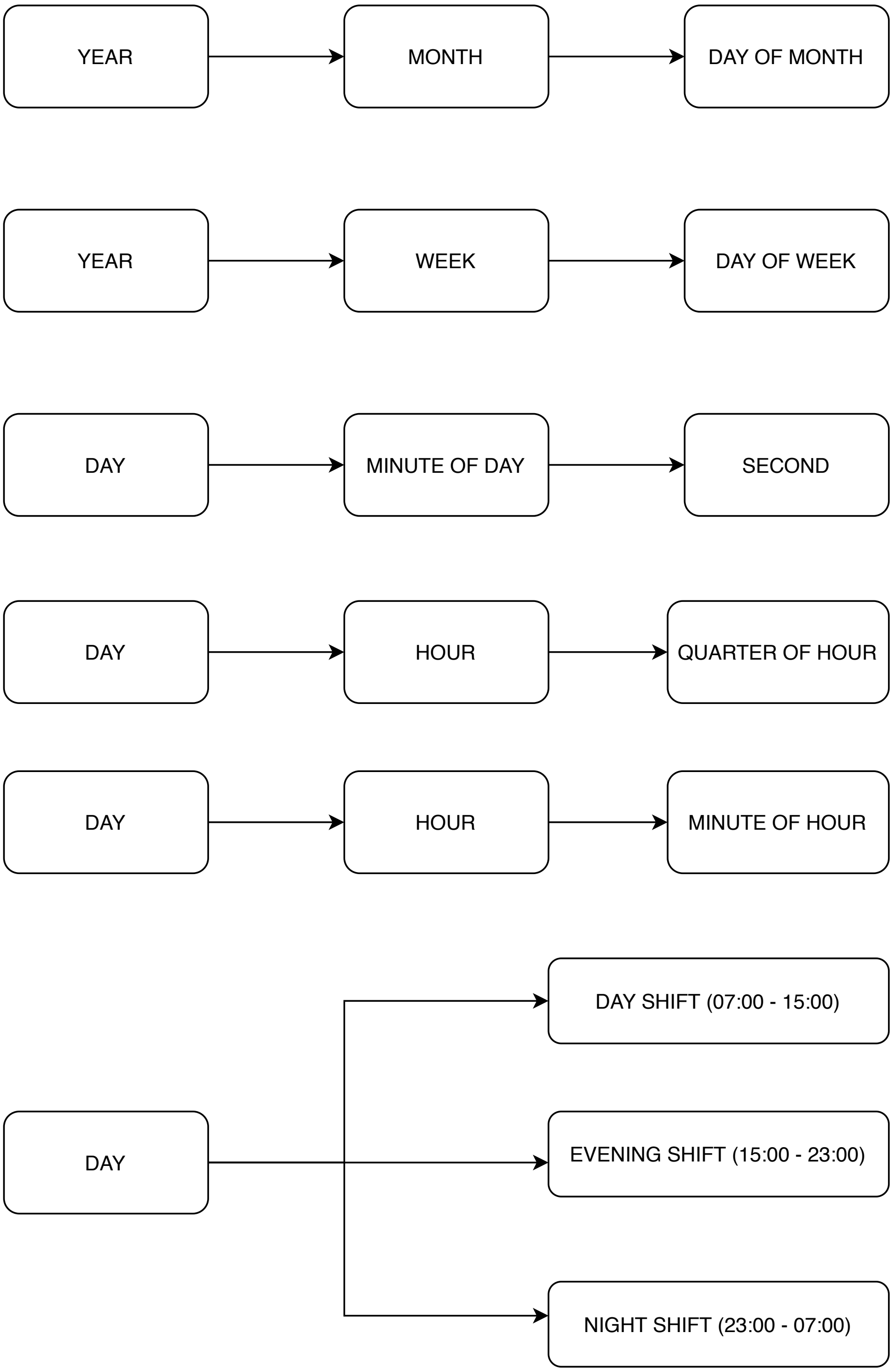


Figure 5. Examples of granularity expressions

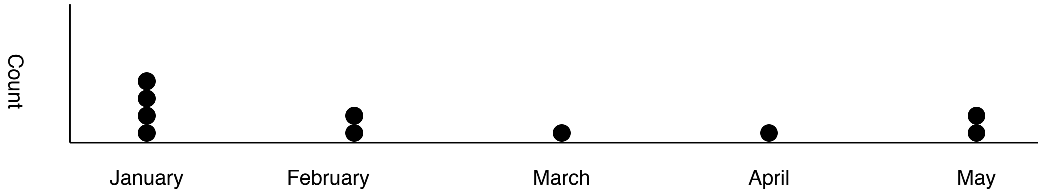


Figure 6. Time points over a period of time

Day Of Month) is a granularity expression, which means that the time granularity *Year* is partitioned to *Month*, which is also splashed to the *Day of Month* in the new subset time granularity. Another example, consider a 24 hours day in a company is partitioned into three periods *morning shift*, *evening shift*, and *night shift*, then each one would be a new granule in the granularity expression.

Definition 2 (Calendar Expression - CE)**.** *A calendar expression (e_n, e_{n-1}, $\dots$, e_1) defines a time point and is described by a respected granularity expression (g_n, g_{n-1}, $\dots$, g_1).*

For instance, CE: $\langle 2020,\ 5,\ 1\rangle$ is a time point calendar expression described by the GE: $\langle Year,\ Month,\ Day\rangle$. It means the first of May, the year 2020. The event log is processed to construct a list of calendar expressions from activities' timestamps. Figure 6 simulates a number of time points that fall into a period between January and May. January, for example, has four calendar expressions, whereas March and April have one each.

Definition 3 (Simple calendar-based Pattern - SP)**.** *A simple calendar-based pattern defines a time point the same as a calendar expression (e_n, e_{n-1}, $\dots$, e_1) except that each granule field in the simple calendar-based pattern presents a time granule domain in the form of an integer or a wild card "*". The wild card "*" presents all possible options of the domain subset.*

For example, $\langle 2020,\ *,\ 1\rangle$ is an SP that follows the granularity $\langle Year,\ Month,\ Day\rangle$. Intuitively, it means the first day of "every" month in the year 2020. Furthermore, a simple calendar-based pattern that does not contain any wild cards is called Basic Time Unit (BTU) according to Yingjiu et al. [7]. A simple calendar-based

Table 1. Set of Calendar Expressions

1	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 3,\ Hour : 3\rangle$
2	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 3\rangle$
3	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 7\rangle$
4	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 6,\ Hour : 6\rangle$
5	$\langle Year : 2012,\ Week : 2,\ DayOfWeek : 1,\ Hour : 6\rangle$
6	$\langle Year : 2012,\ Week : 2,\ DayOfWeek : 1,\ Hour : 14\rangle$
7	$\langle Year : 2012,\ Week : 2,\ DayOfWeek : 2,\ Hour : 3\rangle$
8	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$
9	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$
10	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 5,\ Hour : 13\rangle$
11	$\langle Year : 2012,\ Week : 4,\ DayOfWeek : 2,\ Hour : 2\rangle$
12	$\langle Year : 2012,\ Week : 4,\ DayOfWeek : 3,\ Hour : 3\rangle$
13	$\langle Year : 2012,\ Week : 5,\ DayOfWeek : 5,\ Hour : 13\rangle$
14	$\langle Year : 2012,\ Week : 6,\ DayOfWeek : 1,\ Hour : 13\rangle$
15	$\langle Year : 2012,\ Week : 6,\ DayOfWeek : 6,\ Hour : 2\rangle$
16	$\langle Year : 2012,\ Week : 6,\ DayOfWeek : 6,\ Hour : 3\rangle$

pattern and its granularity expression could be combined into one expression for simplicity as $\langle Year : 2020,\ Month : *,\ Day : 1\rangle$ meaning, the first day of week 8 of the year 2020.

Based on a set of calendar expressions provided as input, the approach proposed in [7] aims to find all simple calendar-based patterns that follow a given granularity expression and satisfy a required predefined constraints. The respected process is referred to as *Calendar-based pattern discovery problem* [7]. The algorithm we implemented to solve the respected matter is to be explained in Section 5.2. The constraints referred to are threshold values recognized as Support and Confidence rates defined below as stated in [7].

Definition 4 (Support rate of a pattern - S)**.** *The percentage of calendar expressions support the presented Simple calendar-based pattern among all input calendar expressions.*

Table 1 has a set of exported calendar expressions following the granularity expression $\langle Year,\ Week,\ DayOfWeek,\ Hour\rangle$. For example, the SP $\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$ is supported by the calendar

Table 2. Calendar Expressions supporting SP: $\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$

1	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$
2	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$

Table 3. Calendar Expressions supporting SP: $\langle Year : 2012,\ Week : 1,\ DayOfWeek : *,\ Hour : *\rangle$

1	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 3,\ Hour : 3\rangle$
2	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 3\rangle$
3	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 7\rangle$
4	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 6,\ Hour : 6\rangle$

expressions in Table 2. The support rate for the respected pattern is therefore $2/16 = 0.125$. For an SP with a wild card "*" such as $\langle Year : 2012,\ Week : 1,\ DayOfWeek : *,\ Hour : *\rangle$, the supporting expressions are demonstrated in Table 3. Support value for the respected pattern equals $4/16 = 0.25$

Definition 5 (Confidence rate of a pattern - C)**.** *Over a time period, the confidence of a simple calendar-based pattern is the percentage of basic time units, among all the basic time unit points given by the simple calendar-based pattern, that contain the given events.*

For example, given the SP: $\langle Year : 2012,\ Week : *,\ DayOfWeek : 4,\ Hour : 7\rangle$ and the data set of calendar expressions in Table 1, it is observed that the period of time the events fall into is between the first week and week six. Therefore, the basic time units yielded by the SP for the respected period are the ones displayed

Table 4. BTUs unfolded by SP: $\langle Year : 2012,\ Week : *,\ DayOfWeek : 4,\ Hour : 7\rangle$ for the period between week 1 and week 6.

1	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 7\rangle$
2	$\langle Year : 2012,\ Week : 2,\ DayOfWeek : 4,\ Hour : 7\rangle$
3	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$
4	$\langle Year : 2012,\ Week : 4,\ DayOfWeek : 4,\ Hour : 7\rangle$
5	$\langle Year : 2012,\ Week : 5,\ DayOfWeek : 4,\ Hour : 7\rangle$
6	$\langle Year : 2012,\ Week : 6,\ DayOfWeek : 4,\ Hour : 7\rangle$

Table 5. CEs contains BTUs by SP: $\langle Year : 2012,\ Week : *,\ DayOfWeek : 4,\ Hour : 7\rangle$

1	$\langle Year : 2012,\ Week : 1,\ DayOfWeek : 4,\ Hour : 7\rangle$
2	$\langle Year : 2012,\ Week : 3,\ DayOfWeek : 4,\ Hour : 7\rangle$

in Table 4. Additionally, Table 5 indicating the BTUs existing in the given CE set 1. As a result, the Confidence of the respected pattern is $6/2 = 3$.

In the following section, we describe the algorithm implementation to discover the resource availability timetable.

5.2 Algorithm Implementation

First, the event log is processed to construct a list of calendar expressions from the events' timestamps according to a predefined granularity expression. We then adopt the approach presented in [7] to discover all simple calendar-based patterns and verify them against the criteria of support and confidence. Next, we construct the resource calendar timetable from the retrieved simple calendar-based patterns. Finally, the timetable is injected into the BPS model discovered by Simod [4], which automates the discovery of the BPS models from event logs. The approach is implemented in the Java programming language and is detailed in the following sections[5].

5.2.1 Granularity Expression Declaration

Granularity Expression is defined for the processing of the event log. The expression determines the extraction level of calendar expressions from timestamps. Please refer to Definition 1 for more input on GE.

The granularity defined and used in this research is **(DayOfWeek, Hour)**. We target discovering a calendar timetable for working hours of each weekday *(e.g., Monday to Friday, from 9:00:00 to 13:00:00 and from 14:00:00 to 17:00:00).* As mentioned before, the level of granularity is adjustable. In other words, higher or

[5] The calendar implementation source code is available at $https : //bitbucket.org/Ibrahim_Mahdy/calendar$.

lower granularity could be defined for the discovery of calendar expressions. For instance, we can discover calendar expressions on the level of minutes "Minute of the day" *(e.g., Monday from 10:15 to 13:05, and from 14:04 to 17:08).* However, we found such a low level affects the readability of the generated timetable.

5.2.2 Event Log Processing

We explore the input log in this stage, going over the given events for the log traces. We extract calendar expressions according to the granularity expression exhibited. A calendar expression is created from a timestamp attached to an event. For example, given timestamp (2020-12-07T10:19:54) and a granularity expression (DayOfWeek, Hour), the respected calendar expression is $\langle DayOfWeek : 1,\ Hour : 10\rangle$, where 1 means the first day of the week (Monday). Furthermore, we obtain the period the event log falls in (the earliest timestamp and latest timestamp found). Exposing the period is crucial for measuring the confidence of a simple calendar-based pattern in the validation stage. Note that pre-defining the period as input could be introduced into the algorithm by specifying a start and end date to explore a specific part of the data instead of processing the entire log period.

5.2.3 Simple Calendar-based Patterns Generation

Different algorithms are presented by Yingjiu et al. [7] to generate simple calendar-based patterns and validate them against the threshold values of the support and confidence. The respected approach follows two steps where they first, generate all possible simple calendar-based patterns, then check the patterns' support and confidence rates.

For the generation of possible SPs, the authors proposed four different algorithms. We adapted the idea of the *Enumeration Algorithm* due to its simplicity and the cleanliness it brings to the code implementation. Also, the enumeration algorithm shows the best performance according to their results. The objective of the Enumeration Algorithm is as follows. Firstly, we extract sets of non-duplicate values, and we call them candidates, from the event log during the log exploration phase. Every set presents a time granule in the granularity expression. For example,

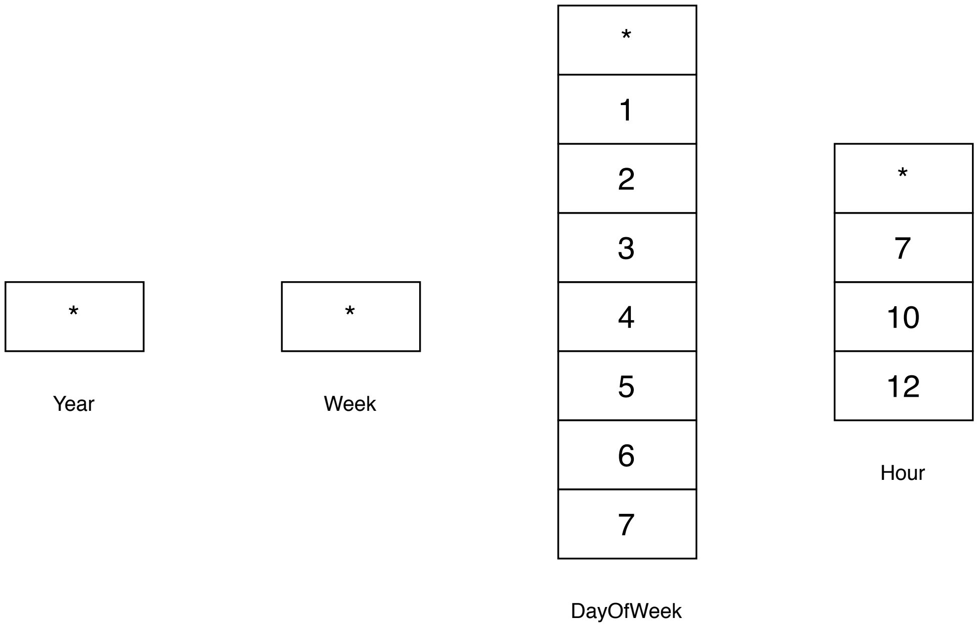


Figure 7. Example of SPs generator candidates

for the time granule “DayOfWeek”, we find the days of the week that have one or more event, resulting in a set that includes a collection of those days, plus the wild card “*”, indicating as mentioned before, all possible varieties (every day).

Secondly, given the explored sets of candidates as input to the Enumeration Algorithm, the output lists all possible simple calendar-based patterns using every possible mix from the input sets. For the sets presented in Figure 7, the generated SPs include: $SP_1 : \langle Year : *,\ Week : *,\ DayOfWeek : *,\ Hour : *\rangle, SP_2 : \langle Year : *,\ Week : *,\ DayOfWeek : 1,\ Hour : *\rangle, ..., SP_{32} : \langle Year : *,\ Week : *,\ DayOfWeek : 7,\ Hour : 12\rangle$. While the total number of created patterns equals the multiplications of all sets’ sizes, in this example, the total count of produced simple calendar-based patterns is thirty-two patterns.

5.2.4 Pattern Validation

Once all simple calendar-based patterns are produced, they are checked against the support and confidence constraints which are provided as input to the algorithm. The output of this stage is a list of simple calendar-based patterns with tight values

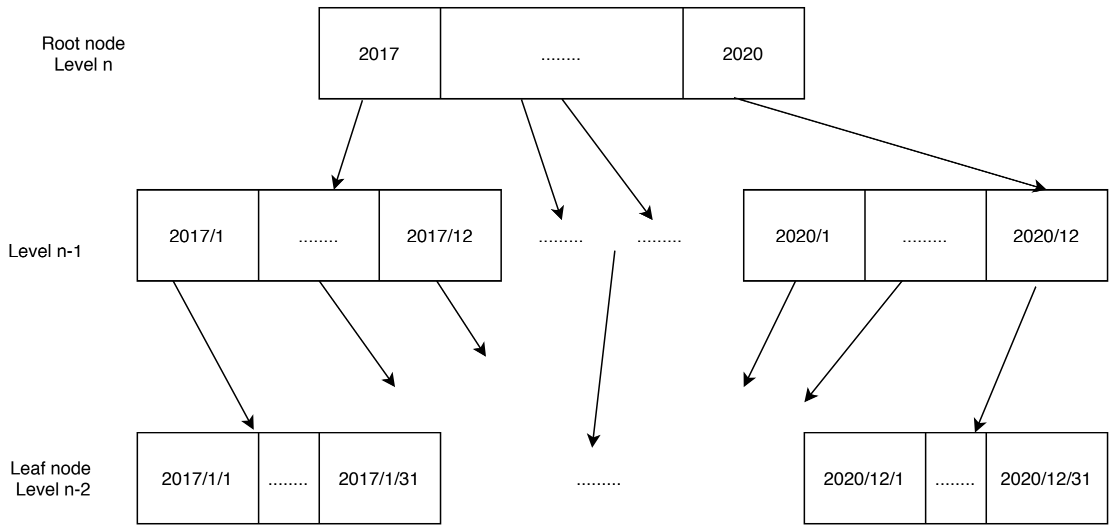


Figure 8. Tree structure [7]

with respect to the entry requirements.

Yingjiu et al. [7], introduced a tree-based calendar structure (see Figure 8) to compute the support and confidence rates of a simple calendar-based pattern. They aim to avoid scanning the entire input every time the count of the calendar expression supporting the respected SP is needed. Hence, a rooted tree is built representing the granularity expression (g_n, g_{n-1}, ..., g_1) with n array levels, one level for every time granule. Each level has a number of nodes that hold all the cells whose corresponding granules are covered by the granule represented by the cell. The highest granularity g_n is constituted by the root nodes, and a node is connected to a lower level of its subset children. In the end, the leaf nodes represent the last granule items in the time granularity g_1. Moreover, they kept a counter for each cell to record the number of elements falling into it. Therefore, the support is computed as the sum of counts in the mapped tree divided by the input size, and the confidence is the number of mapped cells with a counter of more than zero divided by mapped cells' overall count.

By studying the referred tree-based calendar structure, it is found to be over-complicated. Alternatively, our algorithm is based on storing the input calendar expressions exported from the event log in an array list of objects. Every calendar expression has the following attributes *(Year, Month, Week, DayOfMonth, DayOfWeek, Hour)*, more attributes could be added and exported from the respected timestamp if needed. The algorithm's simplicity allows the filtration of the list based on one or more of its object attributes. Therefore, a filtration operation is carried out for a respected simple calendar-based pattern, resulting in a refined listing that includes the SP's supporting expressions. The refined list size and calendar expressions' total volume are used to compute the pattern's support threshold.

The same filtration is done for the SP's confidence computation, except all duplicated items are removed from the produced list. Next, the number of basic time units produced by the SP is calculated where for each wild card symbol in the SP, the respected period is brought into account. For example, consider the SP $\langle\ Year : 2020,\ Month : *,\ Day : 1\rangle$, for a period of twelve months, the total number of BTUs once the SP is unfolded is then twelve elements $[BTU_1$ $\langle\ Year : 2020,\ Month : 1,\ Day : 1\rangle$, BTU_2 $\langle\ Year : 2020,\ Month : 2,\ Day : 1\rangle$, ..., BTU_{12} $\langle\ Year : 2020,\ Month : 12,\ Day : 1\rangle]$. Therefore, the confidence is the size of the filtered set by the number of BTUs generated by the pattern.

It is worth noting that the simple calendar-based patterns generated could be of a tremendous count. For example, take a time granule of seconds for each day of the week within a year. Then the total number of generated SPs would be 1 (Year) * 52 (Weeks) * 7 (Day Of Week) * 86400 (Seconds a day), which comes out totally as 31449600 patterns. An intuitive algorithm would be to loop over every pattern to ensure the requirements' fulfillment, which would result in a performance issue. Consequently, we introduce a pruning technique adopted from [7].

Definition 6 (Intra-level Pruning)**.** *Given a set of data input and a calendar expression e that is a subset of another expression é, if the SP with é does not satisfy the support rate requirement, then the SP with e does not satisfy the support rate requirement.*

The support of an SP equals the sum of all support values of its subset children. Therefore, the support rate of the SP with all wildcards is 100% since it covers every time point in the input dataset. As the granularity level progresses, the

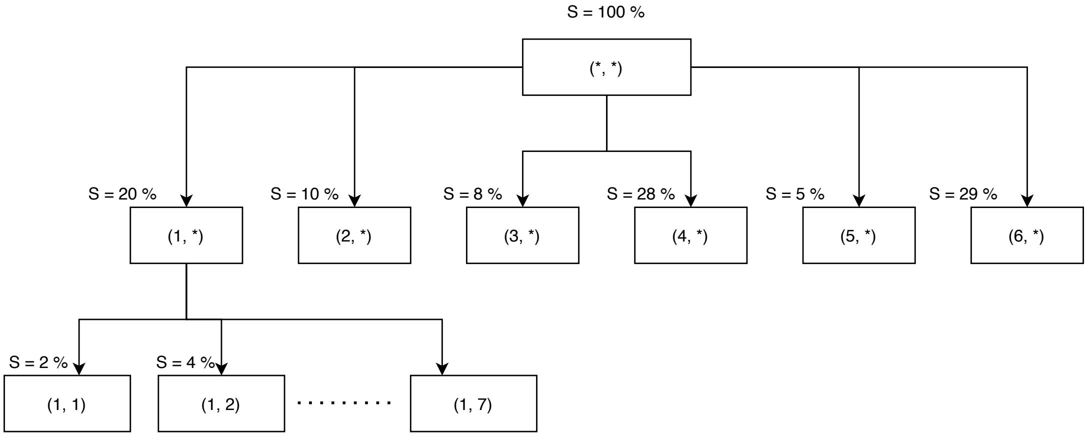


Figure 9. Intra-level pruning

support value drops, divided among the SP children according to the data's support. Hence, if a parent node does not qualify, then all of its children and ancestors do not. Figure 9 demonstrates a tree of patterns, consider that the SP (1, *) does not satisfy the support threshold, then [(1, 1), (1, 2) . . . , (1, 7)] all do not, and we do not need to verify them.

Algorithm 1: Validating simple calendar-based patterns

```
Function validate(SPs: list, sup: double, conf: double) is
    // Initialize the sets of SPs $SP_{satisfied}$ and $SP_{failed}$
    $SP_{satisfied} = \emptyset$ // Simple calendar-based patterns satisfying support and confidence
        requirements
    $SP_{failed} = \emptyset$ // Simple calendar-based patterns failed to satisfy support requirement
        and/or confidence
    for $SP$ generated from input candidates do
        if exists $SP'$ in $SP_{failed}$ then
            discard $SP$
        else
            if support($SP$) $\geq$ sup and confidence($SP$) $\geq$ conf then
                $SP_{satisfied} = SP_{satisfied} \cup \{SP\}$
            else if support(ce) $<$ sup then
                $SP_{failed} = SP_{failed} \cup \{SP\} \cup \{children\}$
            else if support(ce) $<$ conf then
                $SP_{failed} = SP_{failed} \cup \{SP\}$
    return $CE_{kept}$
```

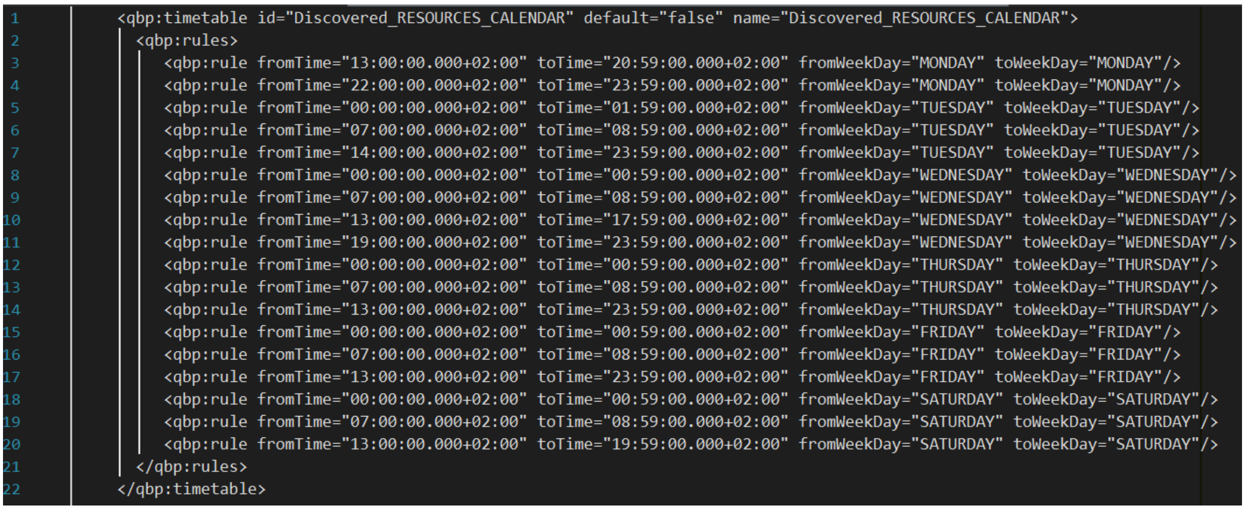
Algorithm 1 describes the steps of patterns validation. We start by initializing two sets; the first to include the SPs satisfying the predefined user requirements of the support and confidence. The second set to add the ones that fail to verify against the thresholds. If the pattern has a good support value, it is checked for the confidence attribute before adding it to the satisfying list. On the other hand, a pattern that breaks the minimum support level is kept in the list of failed patterns alongside its children and ancestors. For an SP passing the support threshold but failing the confidence, it is also added to the failed list. For every pattern, we check first if it falls into the failed list. In such a case, it is discarded before doing any calculations. By the end of the cycle, the list containing the satisfying patterns is returned.

5.2.5 Timetable Construction

The list of simple calendar-based patterns satisfying the support and confidence constraints is used to construct the availability timetable. The hours discovered for each weekday in the event log are formulated according to the syntax required by the BIMP simulator. For example, given that Monday has the following working hours: [13, 14, 15, 16, 17, 18, 19, 20, 22, 23], this easily translates to *fromTime=“13:00:00.000+02:00” toTime=“20:59:00.000+02:00”*, and *fromTime=‘22:00:00.000+02:00” toTime="23:59:00.000+02:00"*. Figure 10 displays an example of the format required by the BIMP simulator. The timetable includes a set of rules, each of which specifies the start and end day and start and end times of the availability. Also, the timetable has three attributes attached to it, a *name* for the timetable, an *id* to identify it since BIMP supports the definition of multiple timetables, and a boolean attribute to determine whether this is the default timetable to take into account when running the simulation.

5.3 Calendar Types

The approach presented in this study allows the extraction of three different calendars. First is the *case calendar*, where every timestamp attached to an event is imported as a time point calendar expression along the process period. The result is an availability timetable covering the entire case of the process. The second is the *resource pools calendar*, which discovers the calendar timetable for



```xml
<qbp:timetable id="Discovered_RESOURCES_CALENDAR" default="false" name="Discovered_RESOURCES_CALENDAR">
  <qbp:rules>
    <qbp:rule fromTime="13:00:00.000+02:00" toTime="20:59:00.000+02:00" fromWeekDay="MONDAY" toWeekDay="MONDAY"/>
    <qbp:rule fromTime="22:00:00.000+02:00" toTime="23:59:00.000+02:00" fromWeekDay="MONDAY" toWeekDay="MONDAY"/>
    <qbp:rule fromTime="00:00:00.000+02:00" toTime="01:59:00.000+02:00" fromWeekDay="TUESDAY" toWeekDay="TUESDAY"/>
    <qbp:rule fromTime="07:00:00.000+02:00" toTime="08:59:00.000+02:00" fromWeekDay="TUESDAY" toWeekDay="TUESDAY"/>
    <qbp:rule fromTime="14:00:00.000+02:00" toTime="23:59:00.000+02:00" fromWeekDay="TUESDAY" toWeekDay="TUESDAY"/>
    <qbp:rule fromTime="00:00:00.000+02:00" toTime="00:59:00.000+02:00" fromWeekDay="WEDNESDAY" toWeekDay="WEDNESDAY"/>
    <qbp:rule fromTime="07:00:00.000+02:00" toTime="08:59:00.000+02:00" fromWeekDay="WEDNESDAY" toWeekDay="WEDNESDAY"/>
    <qbp:rule fromTime="13:00:00.000+02:00" toTime="17:59:00.000+02:00" fromWeekDay="WEDNESDAY" toWeekDay="WEDNESDAY"/>
    <qbp:rule fromTime="19:00:00.000+02:00" toTime="23:59:00.000+02:00" fromWeekDay="WEDNESDAY" toWeekDay="WEDNESDAY"/>
    <qbp:rule fromTime="00:00:00.000+02:00" toTime="00:59:00.000+02:00" fromWeekDay="THURSDAY" toWeekDay="THURSDAY"/>
    <qbp:rule fromTime="07:00:00.000+02:00" toTime="08:59:00.000+02:00" fromWeekDay="THURSDAY" toWeekDay="THURSDAY"/>
    <qbp:rule fromTime="13:00:00.000+02:00" toTime="23:59:00.000+02:00" fromWeekDay="THURSDAY" toWeekDay="THURSDAY"/>
    <qbp:rule fromTime="00:00:00.000+02:00" toTime="00:59:00.000+02:00" fromWeekDay="FRIDAY" toWeekDay="FRIDAY"/>
    <qbp:rule fromTime="07:00:00.000+02:00" toTime="08:59:00.000+02:00" fromWeekDay="FRIDAY" toWeekDay="FRIDAY"/>
    <qbp:rule fromTime="13:00:00.000+02:00" toTime="23:59:00.000+02:00" fromWeekDay="FRIDAY" toWeekDay="FRIDAY"/>
    <qbp:rule fromTime="00:00:00.000+02:00" toTime="00:59:00.000+02:00" fromWeekDay="SATURDAY" toWeekDay="SATURDAY"/>
    <qbp:rule fromTime="07:00:00.000+02:00" toTime="08:59:00.000+02:00" fromWeekDay="SATURDAY" toWeekDay="SATURDAY"/>
    <qbp:rule fromTime="13:00:00.000+02:00" toTime="19:59:00.000+02:00" fromWeekDay="SATURDAY" toWeekDay="SATURDAY"/>
  </qbp:rules>
</qbp:timetable>
```

Figure 10. An example of a timetable for the BIMP simulator.

the resource pools associated with the process. In this case, a list of the resource names of the resource pool has to be provided as input to the calendar discovery algorithm. Therefore, only events in the log carried out by the provided resources are processed, while unrelated events are excluded. Additionally, discovering the calendar of each individual resource instead of a resource pool could be added in the same manner by processing the events performed by the respected resource. Finally, it is also possible to retrieve the *case creation calendar* where the algorithm picks the earliest timestamp recorded for the first event of each trace in the input log. If there is a creation timestamp recorded, it will be chosen to create a calendar expression. Otherwise, the start timestamp is selected. The result is then the calendar for cases' creation or arrival for the process under investigation.

5.4 Simod Integration

Simod automates the generation and validation of BPS models from event logs through various process mining techniques. The calendar discovery algorithm is integrated as an external sub-process plugin. The collaboration was made with the author of Simod to carry out the integration[6]. The plugin is executed

[6] Manuel Camargo, a Phd student in Software Engineering at the University of Tartu.

during the discovery stage in conjunction with other discovery algorithms, for example, retrieving resource pools, branch probabilities, and inter-arrivals times. Furthermore, Simod fine-tunes the support and confidence thresholds required for the calendar algorithm along with other parameters using a Bayesian hyperparameter optimizer. The optimizer explores the search space made up of the combination of all the discovery algorithms' parameters.

The next chapter evaluates the presented approach and presents an analysis of the results.

6 Evaluation

It has been affirmed before that adding the resource availability timetable to the business process simulator will improve the accuracy of the simulation results. In this chapter, the proposed calendar algorithm is evaluated to determine if the proposed claim holds.

6.1 Experimental Procedure

The device used for the evaluation runs the Windows operating system, version 10, Enterprise edition. It has a CPU 1.6GHz, processor Intel Core i5 and carries 16 GB of RAM.

The evaluation has been carried out using synthetic and real-life event logs. The calendar algorithm expects each event in an event log to include the three required attributes mentioned before (case identifier, event identifier, and end timestamp). To that end, the assessment is performed using Simod. The tool requires two more attributes, the start timestamp and the resource that carried out the activity. Simod uses the resource attribute to retrieve the process's resource pools. The start timestamp is required along with the end timestamp to discover the processing time of activities and their respective probability distributions.

Experiments have been performed on four real event logs and one synthetic log. The ACR log (Academic Credential Recognition) is extracted from the University of Los Andes, Colombia information system. Purchasing Example log is the synthetic generated by the Fluxicon Disco Tool and provided as open-source. At the same time, the MP (Manufacturing Production) event log is exported from an Enterprise Resource Planning system [6]. The BPI Challenge 2012W (Business Processing Intelligence Challenge) was provided for the participants of the challenge in the year 2012[7]. It includes a loan application process. The "W" means only activities performed by the human resources are included[8]. Finally, the BPI Challenge 2017W log has updated data of the previous one[9]. The log size is substantial to be handled

[7] https://www.win.tue.nl/bpi/doku.php?id=2012:challenge.

[8] https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f.

[9] https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b.

in a considerable time by Simod. It has been filtered to include the events that took place in the period between 06-25-2016 and 10-07-2016.

Log	Log type	Traces count	Events count	Tasks count	Resources count
ACR	Real event log	954	6870	18	561
Purchasing Example	Synthetic event logs	608	9119	21	27
MP	Real event log	225	4953	26	48
BPI Challenge 2012W	Real event log	8616	59302	6	58
BPI Challenge 2017W (filtered)	Real event log	8941	63764	7	113

Table 6. Event logs used in the evaluation procedure

Table 6 shows the used event logs' formation. It is noted that BPI Challenge 2017W holds the largest number of traces with 8941 and the biggest count of events by 63764. In contrast, MP is the one with the least number of traces and events. On the other hand, ACR has the largest number of resources, with 561 resources to perform 18 tasks. Purchasing Example, however, has as few resources as 27. The BPI Challenge 2012W log is the one with only six tasks to be carried out by 58 resources.[10]

The assessment is objectively carried out between the event log generated by Simod during the simulation stage and the ground-truth event log used as input. The similarity assessment is measured between the two process traces using distance measurements. For instance, the mean absolute error (MAE) is used in fields to calculate how much error there is between two examinations running under the same conditions. Our experiment uses MAE to estimate the duration, in seconds, of the cycle or execution times among traces. Comparison is made between the ground truth event log and the log generated by Simod from the discovered process model.

[10] Event logs, as well as the source code, are available at $https : //bitbucket.org/Ibrahim_Mahdy/calendar$

Moreover, Simod introduces another distance metric called the Timed String Distance (TSD). TSD is a modification of the Damerau-Levenshtein (DL), a quantitative evaluation metric used in predictive process monitoring to check for the similarity between two process traces. TSD, on the other hand, adds a penalty based on the difference in duration in the processing and the waiting times.

Furthermore, the Log Mean Absolute Error (LMAE) is used for evaluation. For two event logs, including the same number of traces, LMAE is used to check the global times' distance between them. In other words, it is the absolute mean error of the time windows of both event logs. The time window is the difference between the last event's complete time stamp and the first event's start timestamp in the process. Consequently, LMAE throws the light on the times in the event log affected by resource availability restrictions.

The final metric is the Earth Mover's Distance (EMD). EMD is another known distance measure used to calculate the distance between two probability distributions over a defined region. For two event logs (Ground-truth and simulated), the normalized histograms of the events in different time windows are compared. The time windows are calculated according to the most frequent circadian cycles of resources. Therefore, We used EMD on the event level to evaluate the temporal dynamics of the logs. The EMD values start from zero, meaning the two logs' observed times are identical and increased to one as the difference rises.

We carried out three different scenarios to achieve the evaluation criteria. Two of the phases make use of the calendar discovery algorithm, while one excludes it. We explain the different scenarios below[11].

Baseline case. When Simod does not make use of the calendar discovery algorithm, it discovers a 24/7 calendar timetable to be used for calendar availability. Moreover, Simod hyper-parameter optimizer is used to find the best combination of parameters for the process configuration running 30 iterations. Each resulting simulation model is then executed five times, generating event logs with the same size as the original

[11] The evaluation procedure and results have been carried out for the article "Discovering Business Process Simulation Models in the Presence of Multitasking and Availability Constraints" currently under review for publication in a scientific journal.

event log. The TSD evaluates the models' accuracy as the distance between the simulated event log and the ground-truth log. Each generated trace is aligned with the original log's similar trace, then the TSD between them is measured.

Additionally, Simod makes use of the best-found timetable by the resource availability calendar presented in this paper. We carried out two different scenarios, namely, Global Improvement (GI) and Local Improvement (LI).

Global improvement case. Simod hyper-parameter optimizer is used to discover optimized parameters as before, except in this case, we also discover the support and confidence values required by the calendar discovery algorithm alongside the other configurations. As a result, the optimizer search space is expanded, requiring more iterations, 50 rather than 30 in the previous phase. Five simulation turns were executed for each event log and configuration to be evaluated by TSD measure as the baseline case.

Local improvement case. In this case, the configuration values are obtained in the same way as the baseline scenario. Then the optimizer tries to locate the best support and confidence parameters for the calendar discovery algorithm. The number of iterations by the optimizer is 30, the same as the baseline. Five simulation turns were executed for each event log and configuration as well. Regarding the accuracy loss functions, instead of using the TSD metric, the EMD is measured with a weekday/hour time window. The reason is that TSD relies on relative timestamps, which makes it suitable for discovering attributes such as the distribution of task duration. EMD, on the other hand, makes use of absolute timestamps, the same as the calendar discovery algorithm.

6.2 Results and Discussion

Results obtained by performing the pre-described analysis cases are graphed in Figure 11 and stated in Table 7. The general trend is evident that the GI and LI have better outcomes than excluding the calendar algorithm. We go over each measure metrics below.

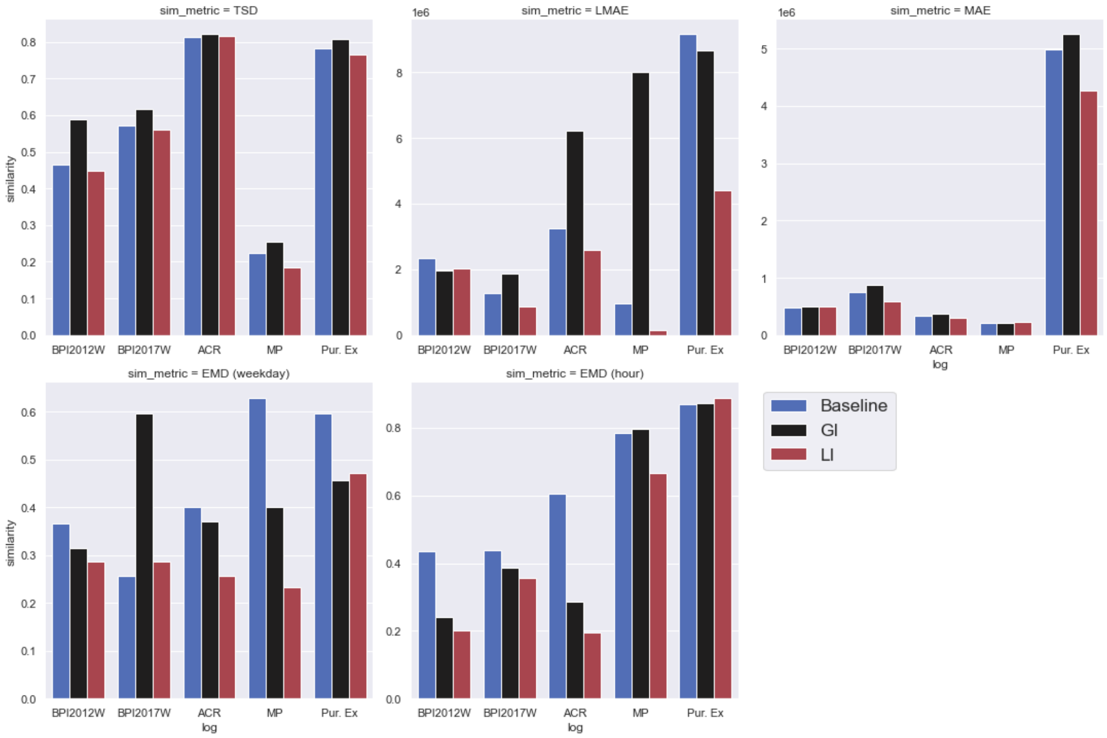


Figure 11. Accuracy results. MAE and LMAE in seconds

In respect of MAE. The LI and GI approaches outperformed in four of the logs, i.e., ACR, Purchase Example, MP, and BPI2017W, and remains very close in BPI2012W. The reason is that the cycle times of traces are evident in these logs, which leads to the reduction of the mean absolute error between the simulated and the ground-truth log.

In respect of LMAE. The LI approach show improvement in four event logs. The MP, in particular, has a significant drop in the LMAE, which reflects two aspects: The first is that introducing the calendar discovery algorithm positively affects the generation of instances in the simulation. Secondly, the resource access conflicts within the process due to the resource being shared by several instances have no factor in the instance's cycle times. The other three logs, i.e., ACR,

Log	**Scenario**	**TSD**	**LMAE**	**MAE**	**EMD (weekday)**	**EMD (hour)**
ACR	**Baseline**	0.8142	3258874	338802	0.4001	0.6062
	GI	**0.8204**	6209179	369895	0.3714	0.2871
	LI	0.8163	**2602998**	**296616**	**0.2572**	**0.1956**
PE	**Baseline**	0.7811	9156301	4993402	0.5953	**0.8678**
	GI	**0.8073**	8674894	5250932	**0.4572**	0.8725
	LI	0.7662	**4391892**	**4269479**	0.4714	0.8875
MP	**Baseline**	0.2249	951211	209036	0.6286	0.7829
	GI	**0.2538**	8004078	**208856**	0.4000	0.7957
	LI	0.1844	**156314**	223114	**0.2334**	**0.6642**
BPI2012W	**Baseline**	0.4637	2339552	**481631**	0.3667	0.4335
	GI	**0.5873**	**1978978**	494508	0.3143	0.2400
	LI	0.4485	2015657	492679	**0.2857**	**0.2000**
BPI2017W	**Baseline**	0.5707	1271119	750593	***0.2572***	0.4370
	GI	**0.6155**	1874672	867037	0.5953	0.3854
	LI	0.5615	**870689**	**591604**	0.2857	**0.3566**

Table 7. Accuracy evaluation results

Purchase Example, and BPI2017W, have clear timeframes that improved precision. BPI2012W, on the other hand, shows the best results by the GI approach, which is still close to LI. This behavior could be explained as a result of the nature of the log. For example, the process may not present actual restrictions on the availability of resources or the instances' creation.

In respect of TSD. TSD shows that GI is the best performing technique, which means that the distance measure between the ground-truth log and the simulated one is enhanced. In other words, the generated simulation model's accuracy is highly improved at the event log's trace level.

In respect of EMD. The local improvement approach enhanced the precision of the generated models in three event logs. The trend is undeniable in the event logs ARC and MP since they have the nature of the circadian process, apprehended by the calendar discovery algorithm. (see Figures 11 - EMD (Weekday) and (hour)).

Though the proposed method shows good potentials, we recognize a few limitations:

1. Defining lower granularity, as low as minutes of the day, leads to many short unavailability time gabs within the working day, making the timetable hard to read. Additionally, performance concerns arise due to processing a large number of simple calendar-based expressions.

2. While the used approach is based on the temporal patterns of events, it is not the only determining factor for resource availability. For instance, contextual behaviors, such as a colleague's illness, will influence the resource schedule and should be captured.

3. The simulator processes work items in the order they arrive (first come, first served) by assigning them to an available resource, while in reality, this assumption does not hold. A resource could process a certain task before another or decides not to perform a particular task and/or batch several ones to carry them out at once. Such considerations could be addressed by dedicating further research to retrieve timetables that are closer to reality.

In this chapter, we evaluated the resource availability algorithm presented in this thesis. We analyzed the results and settled how it improves the generated BPS model's accuracy. We also addressed some of the limitations of our approach. In the next chapter, we conclude our work.

7 Conclusion

In this research, a method has been introduced and evaluated to automatically discover and retrieve resource availability timetable from an event log. The technique was incorporated to improve business process simulation tools that do not consider resource availability constraints. Our approach adopted the discovery of temporal patterns from a set of time points introduced in [7]. The time points are assembled from events' timestamps, creating calendar expressions as input to the algorithm. The algorithm then determines temporal patterns based on predefined granularity expressions. Finally, the discovered patterns construct the availability calendar timetable. By using our proposal, several resource calendar types could be discovered, including case calendar, resource pool calendar, and case creation calendar. We integrated the calendar discovery algorithm with Simod, a data-driven simulation tool that automates BPS models' discovery from event logs. Simod used the calendar discovery algorithm as an external plugin during the BPS model discovery stage. The empirical results and analysis showed that the discovered BPS models' accuracy improved after using the calendar algorithm's discovered timetables. The local improvement approach showed the most promising results by fine-tuning the support and confidence thresholds required for the calendar algorithm after optimizing other parameters for the base simulation model. LI approach also showed that using EMD, a technique that relies on absolute timestamp, as loss function leads to more solid precision.

Future work could be directed to address the limitations mentioned before. For example, lower granularity could be explored and evaluated accordingly, e.g., retrieving the availability by minutes of the day. The behavior of resources could be approached to consider, for example, task prioritization, batching (grouping tasks to perform them at once), resources performance level as different resources have different speed carrying out the same task. Finally, expanding the horizon for resource availability, such as discovering public holidays and vacation patterns.

Acknowledgments. The research work leading to this Masters thesis was funded by the European Research Council (PIX project).

References

[1] Thomas Allweyer. *BPMN 2.0: introduction to the standard for business process modeling.* BoD–Books on Demand, 2016.

[2] Adriano Augusto, Raffaele Conforti, Marlon Dumas, and Marcello La Rosa. Split miner: Discovering accurate and simple business process models from event logs. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1–10. IEEE, 2017.

[3] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24:2546–2554, 2011.

[4] Manuel Camargo, Marlon Dumas, and Oscar González Rojas. Simod: A tool for automated discovery of business process simulation models. In *Proceedings of Demonstration Track - BPM 2019*, pages 139–143, 2019.

[5] Marlon Dumas, Marcello La Rosa, Jan Mendling, and Hajo A Reijers. *Business process management.* Springer, 2013.

[6] Dafna Levy. Production analysis with process mining technology. *Dataset*, 2014.

[7] Yingjiu Li, Xiaoyang Sean Wang, and Sushil Jajodia. Discovering temporal patterns in multiple granularities. In *Proceedings of International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining-Revised Papers*, TSDM '00, pages 5–19, 2001.

[8] Abel Madis. Lightning Fast Business Process Simulator. Master's thesis, University of Tartu, 2011.

[9] Niels Martin, Benoît Benoît, and An Caris. Event log knowledge as a complementary simulation model construction input. In *2014 4th International Conference On Simulation And Modeling Methodologies, Technologies And Applications (SIMULTECH)*, pages 456–462. IEEE, 2014.

[10] Niels Martin, Benoît Depaire, and An Caris. The use of process mining in business process simulation model construction. *Business & Information Systems Engineering*, 58(1):73–87, 2016.

[11] Niels Martin, Benoît Depaire, An Caris, and Dimitri Schepers. Retrieving the resource availability calendars of a process from an event log. *Information Systems*, 88:101463, 2020.

[12] Mahsa Pourbafrani, Sebastiaan J. van Zelst, and Wil M. P. van der Aalst. Supporting automatic system dynamics model generation for simulation in the context of process mining. In *23rd International Conference on Business Information Systems (BIS)*, volume 389, pages 249–263. Springer, 2020.

[13] Daniel Reißner, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, and Abel Armas-Cervantes. Scalable conformance checking of business processes. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 607–627. Springer, 2017.

[14] Daniel Reißner, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, and Abel Armas-Cervantes. Proconformance 2.0 (automata-based). 2018.

[15] Andreas Rogge-Solti, Arik Senderovich, Matthias Weidlich, Jan Mendling, and Avigdor Gal. In log and model we trust? a generalized conformance checking framework. In *International Conference on Business Process Management*, pages 179–196. Springer, 2016.

[16] Anne Rozinat, Ronny S Mans, Minseok Song, and Wil MP van der Aalst. Discovering simulation models. *Information systems*, 34(3):305–327, 2009.

[17] Toma Rusinaite, Olegas Vasilecas, Titas Savickas, Tadas Vysockis, and Kestutis Normantas. An approach for allocation of shared resources in the rule-based business process simulation. In *Proceedings of CompSysTech 2016*, pages 25–32, 2016.

[18] Minseok Song and Wil MP Van der Aalst. Towards comprehensive support for organizational mining. *Decision Support Systems*, 46(1):300–317, 2008.

[19] Wil Van Der Aalst. Data science in action. In *Process mining*, pages 3–23. Springer, 2016.

[20] Wil MP van der Aalst. Business process simulation revisited. In *Workshop on Enterprise and Organizational Modeling and Simulation*, pages 1–14. Springer, 2010.

[21] Wil MP Van der Aalst. Business process management: a comprehensive survey. *International Scholarly Research Notices*, 2013, 2013.

[22] Wil MP Van der Aalst. Extracting event data from databases to unleash process mining. In *BPM-Driving innovation in a digital world*, pages 105–128. Springer, 2015.

Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Ibrahim Mahdy**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Mining Resource Availability for Data-driven Business Process Simulation,

   supervised by Marlon Dumas and Bedilia Estrada-Torres.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Ibrahim Mahdy Yousef
January 13, 2021