Prosimos: Discovering and Simulating Business Processes With Differentiated Resources

Orlenys López-Pintado, Iryna Halenok, and Marlon Dumas

University of Tartu, Tartu, Estonia
{orlenyslp, iryna.halenok, marlon.dumas}@ut.ee

Abstract. Prosimos is an open-source tool that discovers business process simulation models from execution data (event logs) and that enables users to perform what-if analysis using the resulting models. Prosimos distinguishes itself from other data-driven business process simulation approaches in the way it models resources. Existing data-driven simulation approaches treat resources as undifferentiated entities, grouped into resource pools, and assume that all resources in a pool have the same performance and availability calendars. In contrast, Prosimos allows resources (within a pool) to have different performance and availability profiles. For example, instead of treating all claims officers in an insurance claims handling process as having the same performance and availability, Prosimos may capture scenarios where senior resources perform some tasks faster than junior ones, or scenarios where some resources work part-time. To this end, Prosimos integrates algorithms for discovering differentiated resource profiles from event logs.

1 Introduction

Business Process (BP) simulation engines allow users to predict how changes in a process can impact its performance [1]. A BP simulation engine takes as input a simulation model, which usually takes the form of a process model (e.g., a diagram in the Business Process Model and Notation – BPMN), enhanced with parameters such as resource availability, activity processing times, inter-arrival rates of new process cases and branching probabilities at decision gateways in the process model. Given such a model, a BP simulation engine produces an event log of a simulated run of the process, as well as aggregate performance metrics, enabling users to compare multiple what-if simulation scenarios.

Traditional simulation approaches treat resources in a process as undifferentiated entities [1, 3, 5]. These approaches group resources into disjoint resource pools, such that all resources in a pool share the same performance and availability. In these approaches, each activity is assigned to one resource pool. This undifferentiated resource modeling approach implies that the processing time of an activity does not depend on the resource that performs it. Similarly, the undifferentiated treatment of resources implies that all the resources in a pool are available for work during the same periods. In practice, though, each (human) resource exhibits different performance and availability. The assumption that

all resources in a pool behave in the same way, introduces approximations that ultimately have an impact on simulation accuracy [1, 2, 5].

In this demonstration paper, we introduce Prosimos: an open-source simulation tool that implements an approach to BP simulation with differentiated resources. In Prosimos, resources are not grouped into pools but treated as individuals, each with its own resource profile. In particular, the performance of each resource is independent of that of other resources (differentiated performance), and each resource may have its independent availability calendar (differentiated availability). Unlike classic BP simulation models, an activity in a process model may be assigned to multiple resource profiles, and multiple resources can share the same resource profile. The latter also allows us to answer questions like "what if a resource is replaced by another with lower performance?" or "what if a resource changes its availability from full-time to part-time?", not supported by models with undifferentiated resources.

A simulation run in Prosimos produces as output the event log of the simulation, as well as the following performance metrics: mean *waiting time* – the duration from the moment activity is enabled until it is started; mean *processing time* – the duration between the beginning and end of an activity instance; mean *cycle time* – the difference between the end time and start time of a process case; and *resource utilization* – the ratio of the available time of a resource spent executing process activities.

In addition to providing a simulation engine, Prosimos embeds a module to automatically discover simulation models (with differentiated resources) from event logs. This demo paper focuses on the tool architecture and functionality. The description of the underpinning algorithms (including the simulation model discovery algorithms) and other technical details are reported in [6].

2 Prosimos Architecture

Prosimos is a Web-based tool, logically structured into three layers, as shown in Fig. 1. The layer at the bottom, henceforth referred to as Prosimos backend, consists of three groups of components labeled as `SIMOD-DiFF`, `Prosimos Engine` and `Support Modules`. The `Support Modules` consists of a set of components containing supplementary functionalities shared by both the `SIMOD-DiFF` and `Prosimos Engine`. Among those, the `Simulation Model Parser` contains functions for transforming the input files, e.g., BPMN model, XES or CSV event logs, and JSON simulation parameters, into the data structures required by Prosimos. The `BPMN Replayer` implements the token game as specified in the BPMN standard, which serves to execute an event log over the process model, e.g., to estimate branching probabilities and to compute the process state during the simulation. The `Timetables Manager` is a module enclosing the calendar-based operations used for scheduling the inter-arrival time intervals and resource availability. Finally, the `Stochastic Estimator` provides a set of operations to determine and evaluate probability density functions.

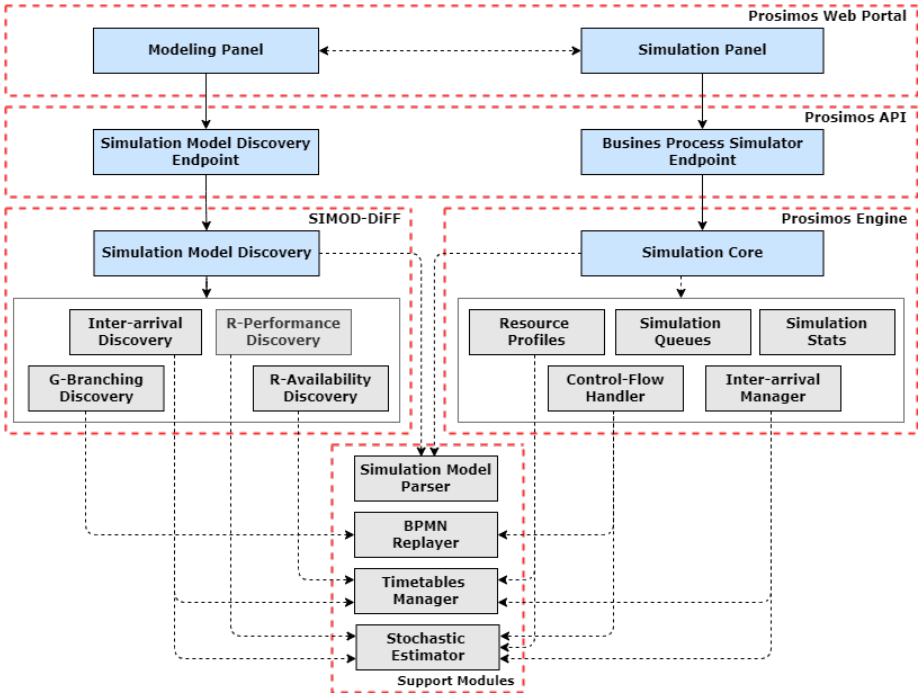

Fig. 1. Prosimos Architecture

On the top-left of the back-end, the `SIMOD-DiFF` components discover the simulation parameters given a BPMN model and the corresponding event log written in XES or CSV, respectively. Precisely, the `Inter-arrival Discovery` module estimates the inter-arrival distribution functions adjusted to an arrival calendar, i.e., how often new cases are created and in which time intervals. The `G-Branching Discovery` calculates the branching probabilities once the execution flow arrives at a decision split gateway, i.e., by replaying the input event log over the process model. The `R-Performance Discovery` estimates the probability density functions modeling how long it would take for each resource to execute its allocated tasks. Finally, the `R-Availability Discovery` retrieves the calendars in which each resource is available to perform a task in the process. As a result, the `Simulation Model Discovery` joints and retrieves all the simulation parameters discovered from the event log into a JSON file.

Concluding with the back-end, the `Prosimos Engine` components handle the business process simulation from a given model. The `Simulation Core`, as the name suggests, is the spine of the simulation, engaging the operation of the remaining five modules to produce, from a simulation model received as input, the corresponding simulation log, and performance indicators as output. The `Resource Profiles` handle the differentiated resources, e.g., including resource allocation, performance (according to the `Stochastic Estimator`), and availability (interacting with the `Timetables Manager`). The `Control-Flow Handler` interacts with the `BPMN Replayer` and the `Stochastic Estimator` to

Verb	URI	Description
POST	/api/discover	Discovers the simulation parameters given a BPMN model and an event log
POST	/api/simulate	Performs the simulation from a given simulation model
GET	/api/results	Retrieves the event logs and metrics produced as result the simulation

Table 1. Prosimos REST API

compute the state of each simulated process case, i.e., the activities enabled at each moment of the simulation, and to decide which path to follow at each split decision gateway. The `Inter-arrival Manager` uses the inter-arrival distribution functions (`Stochastic Estimator`) and arrival calendar (`Timetables Manager`) to create new process instances to simulate. Prosimos uses a priority multi-queue data structure, in `Simulation Queues`, that handles shared activities and sorts resources according to an allocation input function, i.e., according to resource availability as a default. Finally, the `Simulation Stats` computes the performance indicators of the simulation.

The two layers on top of the architecture, i.e., the Prosimos front-end, consist of the `Prosimos API` and `Prosimos Web Portal`. The Prosimos REST API, in the middle of the architecture, provides three endpoints, grouped into the `Simulation Model Discovery Endpoint` and `Business Process Simulator Endpoint`. Table 1 describes the verbs, URIs, and actions of those endpoints. Although the REST API can span more specific endpoints, e.g., for interacting with the back-end to trigger the `BPMN replayer` or discover calendars for a given resource, for simplicity, we kept the API with the minimum operations required to discover simulation models and to run simulations. On top of the architecture, the `Modeling Panel` provides a web interface for end-users to create, modify or discover (interacting with the Prosimos API) simulation models from event logs. On the right, the `Simulation Panel` allows users to run simulations and retrieve the resulting event logs and performance metrics.

3 Source Code, Evaluation and Demonstration Screencast

End users can discover and simulate business processes with differentiated resources via the software-as-a-service deployment of Prosimos at `https://prosimos.cloud.ut.ee`. A sample process model and an event log are available at `https://shorturl.at/abrs0`. The Prosimos Web application can be deployed via a Docker container by using the scripts available at `https://github.com/AutomatedProcessImprovement/prosimos-docker`. This code repository also includes links to the Prosimos Web Portal and REST API code repositories. The source code of Prosimos back-end can be downloaded from `https://github.com/AutomatedProcessImprovement/Prosimos`. The code distribution provides the `SIMOD-DiFF` components described in the architecture in the folder `bpdfr_discovery`. The `Prosimos engine` and `Support Modules` are in the folder `bpdfr_simulation_engine`.

Prosimos is in its first release and supports the following elements in the standard BPMN 2.0: default start and end events, tasks, inclusive, exclusive,

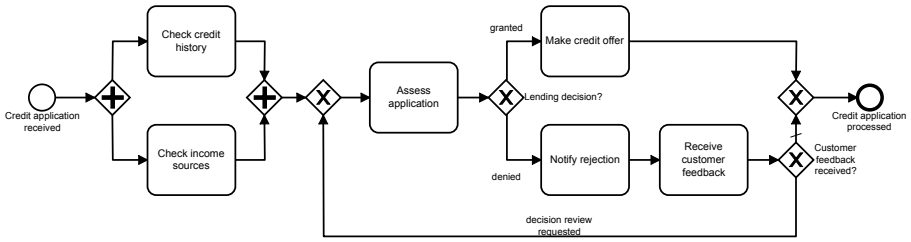

Fig. 2. Simplified credit application process model

and parallel gateways. Specifically, in the case of the exclusive gateway, Prosimos implements the complete OR join semantic as prescribed by the BPMN 2.0 standard. The selection of those elements in the first release aligns with the output of the existing approaches to discovering BPMN models from event logs. The latter is not a significant limitation as data about other advanced BPMN elements, e.g., throwing/interrupting events, event-based gateway, messages, data objects, and multi-instance attributes, are usually not present in event logs. Thus, they are typically not present in process models discovered automatically from data. Still, we are working to introduce most of those advanced BPMN elements in the next release of the Prosimos simulation engine so that process analysts can simulate processes with a broader spectrum of BPMN. The last updates about the models supported by Prosimos can be accessed from `https://shorturl.at/dgnsv`.

Prosimos implements the simulation approach with differentiated resources presented and evaluated in the research paper [6]. We empirically assessed the discovered models by simulating them using Prosimos and measuring the distance between the simulated logs and the original ones. Specifically, in line with [4], we compare simulated and original event logs by extracting temporal histograms from each event log and computing the Earth Movers' Distance (EMD) between these histograms. The experiments in [6] show that simulation models with differentiated resources produce simulated event logs closer to the original event logs when compared to equivalent models with undifferentiated resources. For further details about the evaluation, we refer the reader to check [6]. Additionally, the folder `testing_scripts` in the Prosimos back-end repository includes all the scripts to run the experiments presented to assess the approach. Finally, some components of Prosimos, like the `BPMN replayer` and the `Timetables Manager`, have been integrated and tested into Simod [3], which is a well-known tool in the BPM community to discover undifferentiated models.

For the demo, we will use the simplified model of a credit application process shown in Fig. 2. Two files must be provided to simulate the process: (i) the process model specified in the BPMN standard and (ii) the simulation parameters written in JSON format. Alternatively, the JSON file with the simulation parameters can be discovered from an event log written in XES or CSV format.

A screencast of Prosimos can be found at `https://shorturl.at/FNPS9`. Specifically, the demo starts with a case in which the simulation parameters are discovered from an event log in XES format, corresponding to the process in Fig. 2. Prosimos includes web templates to manually create and update the sim-

ulation parameters, which are transformed later into the required JSON format. So, once provided the BPMN model and selected the option to specify the simulation parameters, i.e., discovered from a log, loaded from an existing JSON, or created manually from scratch, Prosimos redirects to a view to update the parameters or to start the simulation. We refer the readers to check the Prosimos back-end repository for a more detailed explanation of those simulation parameters, i.e., represented by the tabs: case creation, resource calendars, resources, resource allocation, and branching probabilities.

After running the simulation, Prosimos displays the view simulation results with statistics like the cycle, processing and waiting times, and resource utilization. Business process analysts can use these statistics to detect inefficiencies and decide how to improve their business processes. For example, in the scenario illustrated by the screencast, we can observe the task *Assess Aplication* in Fig. 2 may be a bottleneck due to a very high waiting time. We can also observe (from the discovered simulation parameters) that a single resource executes this problematic task, i.e., the *Credit_Officer_1*, which also works part-time (from 13:00-17:00 on working days). The statistics also show this resource has a very high resource utilization, indicating that the performance of the process may be affected by a resource contention issue. A possible solution would be to add some extra resources or extend the working times of the existing one. Then, after applying the change, the analyst can run a new simulation and quantitatively assess the impact. Additionally, Prosimos produce the simulated event log, which can serve to perform automated analysis and optimization of the process.

The required files to reproduce the demo can be found in the Prosimos docker repository under folder `demo_example` or at `https://shorturl.at/abrs0`. Each of the above-mentioned code repositories provide instructions to install the required dependencies locally.

Acknowledgment: Work funded by European Research Council (PIX project). Iryna Halenok is also supported by the Estonian Ministry of Foreign Affairs – Development Cooperation and Humanitarian Aid funds.

References

1. van der Aalst, W.M.P.: Business process simulation survival guide. In: Handbook on Business Process Management 1, 2nd Ed. pp. 337–370 (2015)
2. Afifi, N., Awad, A., Abdelsalam, H.M.: RBPSim: A resource-aware extension of BPSim using workflow resource patterns. In: Proceedings of ZEUS. pp. 32–39 (2018)
3. Camargo, M., Dumas, M., González, O.: Automated discovery of business process simulation models from event logs. Decis. Support Syst. **134**, 113284 (2020)
4. Camargo, M., Dumas, M., Rojas, O.G.: Learning accurate business process simulation models from event logs via automated process discovery and deep learning. In: Proceedings of CAiSE. pp. 55–71. Springer (2022)
5. Freitas, A.P., Pereira, J.L.M.: Simulation of BPMN process models: Current BPM tools capabilities. In: Proceedings of WorldCIST. p. 557–566. Springer (2016)
6. López-Pintado, O., Dumas, M.: Business process simulation with differentiated resources: Does it make a difference? In: Proceedings of BPM. pp. 361–378. Springer (2022)