Using Event Logs to Model Interarrival Times in Business Process Simulation

Niels Martin[1(✉)], Benoît Depaire[1], and An Caris[1,2]

[1] Hasselt University, Agoralaan – Building D, 3590 Diepenbeek, Belgium
{niels.martin,benoit.depaire,an.caris}@uhasselt.be
[2] Research Foundation Flanders (FWO), Egmontstraat 5, 1000 Brussels, Belgium

Abstract. The construction of a business process simulation (BPS) model requires significant modeling efforts. This paper focuses on modeling the interarrival time (IAT) of entities, i.e. the time between the arrival of consecutive entities. Accurately modeling entity arrival is crucial as it influences process performance metrics such as the average waiting time. In this respect, the analysis of event logs can be useful. Given the limited process mining support for this BPS modeling task, the contribution of this paper is twofold. Firstly, an IAT input model taxonomy for process mining is introduced, describing event log use depending on process and event log characteristics. Secondly, ARPRA is introduced and operationalized for gamma distributed IATs. This novel approach to mine an IAT input model is the first to explicitly integrate the notion of queues. ARPRA is shown to significantly outperform a benchmark approach which ignores queue formation.

Keywords: Business process simulation · Process mining · Interarrival time modelling

1 Introduction

Business process simulation (BPS) refers to the imitation of business process behavior through the use of a simulation model. By mimicking the real system, simulation can identify the effects of operational changes prior to implementation and contribute to the analysis and improvement of business processes [7].

A BPS model is composed of several building blocks such as entities, activities and resources [6]. This work is related to entities, which are dynamic objects that flow through the system and on which activities are executed [2], e.g. passengers when modelling an airline's check-in process. As for each BPS model building block, several modelling tasks are related to entities [6]. This paper focuses on the entity arrival rate, i.e. the pattern according to which entities arrive in the process.

Accurately modelling entity arrival is crucial as it has a major influence on process performance metrics such as the average waiting time or the flow time, i.e. the total time spent in the system. To identify an interarrival time (IAT) input model, i.e. a parameterized probability distribution [3] for the time between the arrival of consecutive entities, inputs can be gathered by e.g. observing the process. However, as process

M. Reichert and H.A. Reijers (Eds.): BPM Workshops 2015, LNBIP 256, pp. 255–267, 2016.
DOI: 10.1007/978-3-319-42887-1_21

observations are rather time-consuming, the presence of more readily available information sources should be investigated. In this respect, process execution information stored in event logs can be useful. Such files, originating from process-aware information systems (PAIS) such as CRM-systems, contain events associated to a case, e.g. the start of a passenger's check-in, where a case is the event log equivalent for an entity. For each event, information is recorded such as the associated activity and a timestamp [12]. This work focuses on the use of process mining, i.e. the analysis of event logs, to support IAT input model specification.

Despite the potential value of event log analysis to model the entity arrival rate, research efforts on the topic are limited. Moreover, they implicitly assume that the first recorded timestamp is the actual arrival of a case, which is not necessarily true. To this end, this paper presents an IAT input model taxonomy for process mining, demonstrating that the latter assumption is only appropriate under particular conditions. When these do not hold, entity arrival times can no longer be directly retrieved from a log as queues are formed for the first activity. Hence, novel modelling methods are required. In this respect, this work presents a new algorithm, called ARPRA, which is the first to integrate the notion of queues when mining an IAT input model.

The remainder of this paper is structured as follows. The following section illustrates the importance of accurate IAT modelling and discusses the scarce related work. The third section presents the aforementioned IAT input model taxonomy. The new algorithm, APRRA, is discussed and evaluated in the fourth and fifth section, respectively. The paper ends with a conclusion.

2 Preliminaries

2.1 Running Example and Problem Statement

Throughout this paper, the check-in process of a fictitious small airline will serve as a running example. The process model is visualized in Fig. 1.

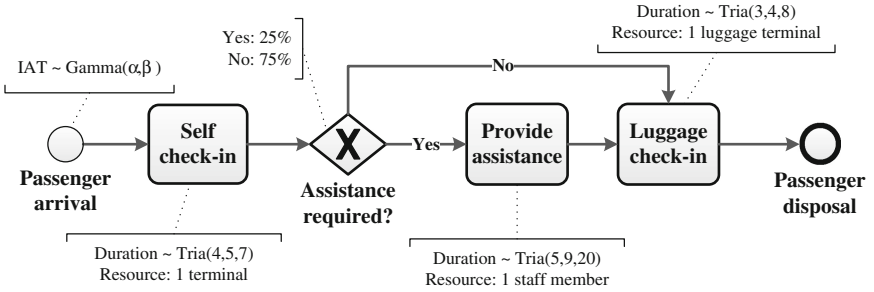

Fig. 1. Running example

A small airline recently started operations at a local airport. To limit staff requirements, the airline installed a self check-in terminal. Arriving passengers follow the terminal's check-in procedure. When assistance is required, they can proceed to the

airline's assistance desk. The final process step is the luggage check-in at the luggage terminal. Based on limited process observations, the company assumes that activity durations follow a triangular distribution and assistance is required for 25 % of the passengers. All assumed parameters are annotated in Fig. 1, with minutes as the time unit. Resource capacities are constant throughout the day.

The running example can be used to show the importance of accurate arrival rate modelling. Suppose passenger IATs are gamma distributed, a more generic distribution than the popular exponential distribution [5], with 1.50 and 5.80 as its shape and scale parameters. Table 1 presents some process performance metrics for alternative parameter values. Note that the process will reach a steady state for each parameter set as the utilization factor ρ is smaller than one for each activity in each scenario [4]. After 30 replications of 12 h, results show that deviations from the assumed IAT input model parameters can have disproportionate effects on performance metrics. E.g. a 10% underestimation of the distribution parameters leads to an overestimation of the average flow time and average waiting time for self check-in of 13.34 % and 70.80 %, respectively. When the simulation study is used to e.g. evaluate the necessity of adding a second self check-in terminal, a flawed IAT input model can lead to inappropriate decisions. This shows the need for accurate entity arrival rate modelling.

Table 1. Effect of inaccurate IAT modelling

Gamma distr. parameter (shape/ scale)	Average flow time	Average waiting time for self check-in	Utilization self check-in terminal
1.50/5.80 (assumed)	18.06	2.74	0.64
1.65/6.38 (+10 %)	15.80 (–12.51 %)	1.39 (–49.27 %)	0.53 (–17.19 %)
1.80/6.96 (+20 %)	14.78 (–18.16 %)	0.76 (–72.26 %)	0.45 (–29.69 %)
1.35/5.22 (–10 %)	20.47 (+13.34 %)	4.68 (+70.80 %)	0.73 (+14.06 %)
1.20/4.64 (–20 %)	32.82 (+81.73 %)	15.81 (+477.00 %)	0.90 (+40.63 %)

2.2 Related Work

Despite the importance of an accurate IAT input model and the fact that event logs typically contain vast amounts of process execution information, thorough research on how to use this information to support entity arrival rate modeling is lacking.

A dotted chart, representing the events of all cases by dots [11] can provide preliminary insight in the arrival rate. However, a mere visual inspection of a dotted chart is insufficient to determine an appropriate IAT input model. The only reference on process mining in a BPS context that briefly mentions arrival rate modelling is Rozinat et al. [9]. These authors calculate IATs as the difference between the first recorded timestamp of two consecutive cases. Afterwards, an a priori assumed exponential distribution is fitted on these IATs.

Both dotted charts and the approach of Rozinat et al. [9] implicitly assume that a case arrives at its first recorded timestamp, which is not necessarily the case. As will be shown in Sect. 3, queues for the first activity can cause entities to have arrived earlier than their first registered timestamp. Despite the fact that queue formation is a common

situation in real-life, research which takes this observation into account when mining an IAT input model is lacking. When an entity's entrance in the first activity's queue is recorded, as is the case in the recently introduced notion of Q-logs [10], this event's timestamp corresponds to entity arrival. However, hypothesizing the presence of a Q-log is a strong assumption. de Smet [1] takes this into account by representing the process as a set of queues based on an event log without queue-related events. However, entity arrival in a particular queue is still equated to the start event timestamp. Consequently, retrieving an IAT input model from a log without queue-related events remains an open challenge, stressing the relevance of this work.

3 IAT Input Model Taxonomy for Process Mining

Defining an IAT input model requires insights in the entity arrival time. This entity arrival time might or might not be directly observable in an event log, depending on the process structure and logging characteristics. When arrival times are directly retrievable, IATs can be calculated from the log and a probability distribution can be fitted. Otherwise, more advanced techniques are required.

To structure the use of process mining in IAT input modeling, Fig. 2 introduces a novel taxonomy. It takes into account four dimensions influencing the IAT modeling approach that should be used: (i) the number of first activities in the process, (ii) whether the first activity involves processing, (iii) whether resource limitations are present and (iv) the logged event types. For the sake of clarity, the numerical references in Fig. 2 will also be used in the discussion below.

When focusing on processes with a single start activity (1), as is the case in the running example, entity arrival times are directly available in the event log in two taxonomy situations. Firstly, the entity arrival timestamp corresponds to the first activity start timestamp when this activity involves no processing (1.1). Moreover, both the start and end timestamp coincide in the absence of processing. Secondly, even when the first activity requires processing, the modeler can proceed to direct IAT calculation when the associated resources have an unlimited capacity and start events are recorded (1.2.1.1, 1.2.1.2). In both aforementioned situations, the implicit assumption made in dotted chart analysis and by Rozinat et al. [9] outlined above is suitable.

In contrast, other taxonomy entries inhibit the exact determination of an entity's arrival time from an event log. Firstly, this is the case when unlimited resources are available, but only end events are recorded (1.2.1.3). The discrepancy between entity arrival and the first recorded timestamp corresponds to the first activity duration. Dealing with this issue is beyond the scope of this work. Secondly, when the first activity requires processing and the associated resources are limited (1.2.2), entities might have arrived earlier than their first recorded timestamp as queues can be formed. In these cases, assuming a correspondence between entity arrival and the first recorded timestamp falsely ignores this notion of queues. Consequently, new methods are required to determine an IAT input model without exactly knowing the moment at which entities arrive, taking into account queue formation. ARPRA, outlined in Sect. 4, is developed in this context.

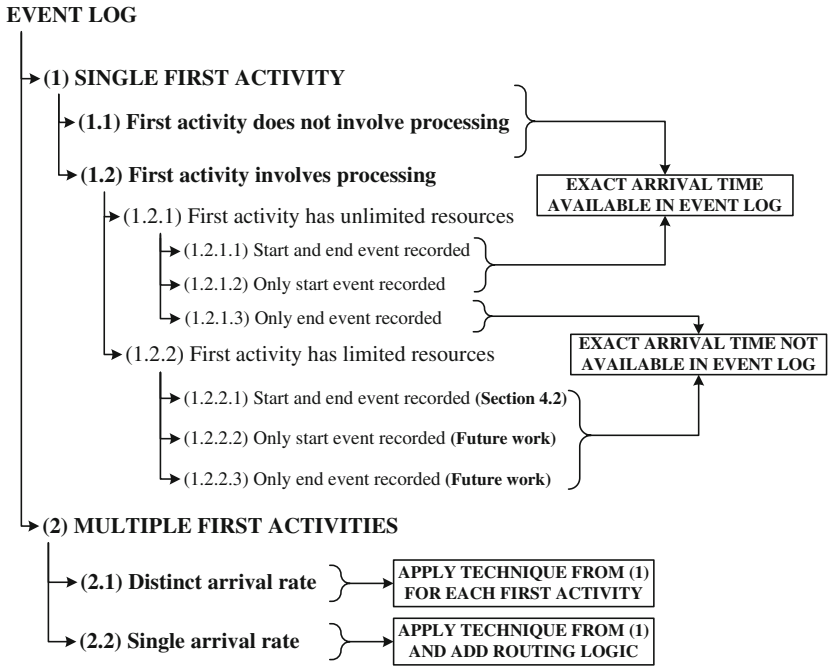

Fig. 2. IAT input model taxonomy for process mining

The prior discussion focuses on processes with a single first activity. In reality, multiple activities can instigate the process (2). Consider e.g. that the airline also develops an online check-in platform as an alternative for the self check-in terminal. Regarding arrival rate modeling, a distinct arrival rate might be specified for each of these first activities or a single IAT input model might need to be defined. When every first activity has its proper arrival rate (2.1), the appropriate technique from the single first activity situation is applied for each of them. Conversely, suppose the modeler wishes to create a simulation model that meets the workflow net requirements, only a single source place and hence IAT input model is allowed [14]. In this case, no distinction is made between the first activities and the appropriate method from case (1) is applied. The single source place should be followed by a decision point determining the first activity for a particular entity.

4 Overview of ARPRA

This section introduces ARPRA, an Arrival Rate Parameter Retrieval Algorithm, which is the first algorithm integrating the queue notion when mining an IAT input model. Queue formation for the first activity renders it impossible to calculate IATs directly from the log as exact arrival times are unknown. This has to be taken into account to avoid a bias in the IAT input model, which stresses ARPRA's contribution.

The first subsection presents the general principles of ARPRA, which are widely applicable as it is e.g. defined independent of the used IAT probability distribution. The second subsection operationalizes the algorithm for gamma distributed IATs.

4.1 Outline of ARPRA

The logic behind ARPRA, as visualized in Fig. 3, can be summarized as follows. Its main input is the proportion of entities that queued upon arrival in the event log (q). Given this percentage, the algorithm iteratively adjusts the parameter set (Ψ) of a particular IAT probability distribution ($\tilde{f}$) until the queue proportion in a simulated log ($\tilde{q}$) matches the queue proportion from the original event log (q). After a pre-specified number of matches (r) is obtained, an aggregated parameter set estimate ($\Psi_{selected}$) is returned.

The remainder of this subsection will outline ARPRA in more detail. Consider a PAIS-supported real-life process with an unknown probability distribution $f(\Psi_{real})$ for the IATs and $g(\theta_{real})$ for the first activity duration (FAD). This process generates an event log, from which three ARPRA inputs are retrieved: the percentage of entities that queued upon arrival (q), knowledge on the first activity duration ($\widetilde{(g(\theta))}$) and an initial estimate for the IAT distribution parameter set (ψ_0). Executable definitions will be provided in Sect. 4.2.

Besides the event log inputs, global parameters are required to use ARPRA. An IAT probability distribution ($\tilde{f}$) needs to be put forward, which will determine the size of the parameter set (Ψ). Other global parameters that need to be specified are the tolerated deviation from the queue proportion in the log (δ), the size of the simulated log in each iteration ($\tilde{n}$), the number of tolerable estimates required to end the algorithm (r) and the number of additional iterations to verify the stability of the queue proportion associated to the recorded tolerable estimates (v).

Based on the above event log inputs and global parameters, ARPRA can mine the IAT input model. In Fig. 3, the rectangle representing ARPRA is subdivided in two parts by a dashed line. The upper part refers to the identification of a series of candidate parameter sets, the lower part reflects final output selection.

Given the IAT probability distribution ($\tilde{f}$) and initial parameter set (ψ_0), an initial IAT input model is obtained. The process is simulated and the queue proportion ($\tilde{q}$) is calculated from a simulated log. When $\tilde{q}$ is outside a tolerance margin δ from the original event log queue proportion (q), the parameter set is adjusted and a new iteration starts. Conversely, when $\tilde{q}$ is between $q - \delta$ and $q + \delta$, the solution of the current iteration is recorded in Φ and iteration continues. Iteration ends when a pre-specified number of parameter set estimates are recorded, i.e. when $|\Phi| = r$.

When r candidate parameter sets are recorded, the lower part of the rectangle in Fig. 3 will select the final output. For each of the r candidate parameter sets, v additional $\tilde{q}$ values are determined and recorded in $\tilde{q}_{list}$ to verify if the initially recorded $\tilde{q}_i$ is representative for parameter set Ψ_i. Given the fact that each simulated log is based on random IAT draws from $\tilde{f}(\Psi)$, different $\tilde{q}$ values can be obtained for the same parameter set Ψ. This can be illustrated using the running example, assuming that passenger IATs follow

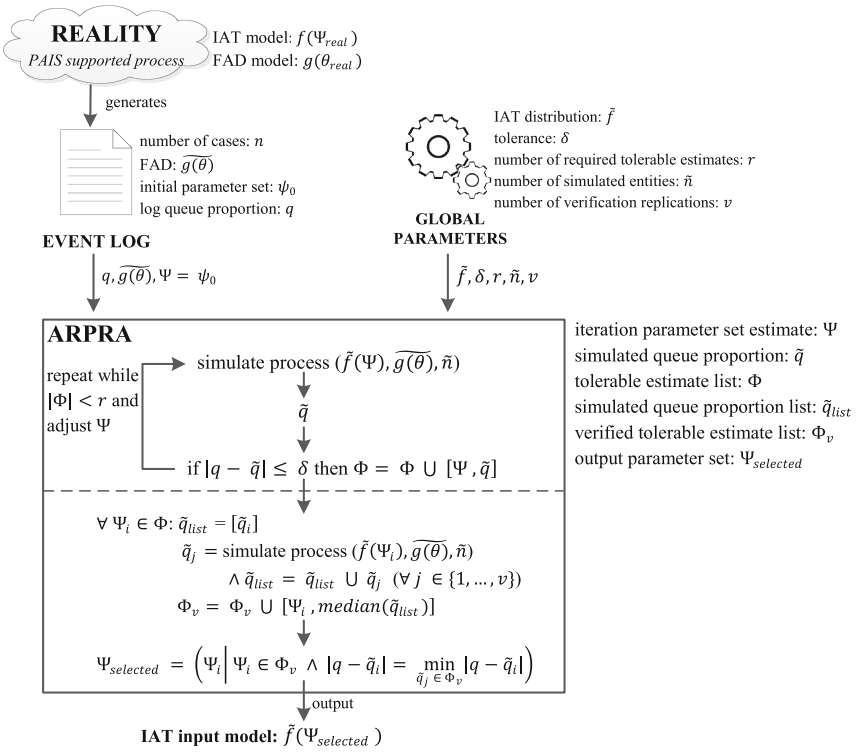

Fig. 3. Overview of ARPRA

a gamma distribution with $\alpha = 1.5$ and $\beta = 5.80$. After generating 2000 simulated logs with the same first activity durations, the obtained $\tilde{q}$ values for these logs range from 39.50 % to 64.25 % with a first and third quartile of 49.69 % and 54.50 %, respectively. This shows the necessity to verify the representativeness of the queue proportion associated to the candidate parameter sets. A verified tolerable estimate list Φ_v is created in which each parameter set from Φ is recorded, together with the median value from its associated $\tilde{q}_{list}$.

From Φ_v, the final output of ARPRA is retrieved, which is the recorded parameter set Ψ that leads to the closest approximation of q. In case of ties, an aggregated parameter set is returned by e.g. calculating the mean.

4.2 ARPRA Operationalization

To evaluate the performance of ARPRA, this subsection outlines an operationalization for situation 1.2.2.1 in the taxonomy presented in Fig. 2. For the sake of clarity, this subsection focuses on the key implementation concepts.

4.2.1 Event Log Inputs

ARPRA requires three key event log inputs, which need to be operationalized to obtain an executable algorithm. Firstly, the main input of ARPRA, the proportion of entities that queued upon arrival (q), is mined by studying the first activity start timestamp of consecutive entities. An entity had to wait when the execution of the first activity started immediately after the first activity is completed for the previous entity. When this is the case, e.g. for passenger 3 in Table 2, the value True is assigned to a boolean *Queue*. Otherwise, this variable is set to False. Once the *Queue* value is determined, q can be determined by dividing the number of cases for which *Queue* equals True by the total number of cases.

Table 2. Illustration of *Queue*-value assignment

Passenger	Self check-in start	Self check-in end	Queue
1	26/05/2015 11:04:28	26/05/2015 11:09:07	False
2	26/05/2015 11:14:55	26/05/2015 11:20:04	False
3	26/05/2015 11:20:04	26/05/2015 11:25:40	True
4	26/05/2015 11:27:42	26/05/2015 11:30:51	False
…	…	…	…

Secondly, initial parameter estimates ψ_0 are determined by fitting probability distribution $\tilde{f}$ on known IAT values in the original event log. IATs are exactly known when two consecutive entities did not queue upon arrival, i.e. have *Queue* = False.

Finally, a trace-driven approach is used regarding first activity durations [8], which refers to the direct use of event log durations when simulated logs are created in ARPRA's iterations. This approach is selected because queue formation is influenced by the interaction between entity arrival and activity duration. As a consequence, ARPRA will use the first activity durations in the same order as observed in reality. When the number of simulated entities ($\tilde{n}$) exceeds the number of entities described in the event log (n), the observed FAD sequence is repeated.

4.2.2 Global Parameters

Values also need to be assigned to ARPRA's global parameters. IAT distribution $\tilde{f}$ is equated to a gamma distribution: a two-parameter distribution with shape parameter α and scale parameter β. When $\alpha = 1$, a gamma distribution corresponds to an exponential distribution [5], which is commonly cited in simulation literature for IAT modeling purposes [2, 5, 8, 13]. The gamma distribution is purposefully selected because it is more generic, but still allows for the popular exponential IAT distribution. Hence, $\Psi = \{\alpha, \beta\}$.

Besides $\tilde{f}$, several other global parameters need to be specified. In the operationalization, the queue proportion tolerance margin $\delta = 0.01$, the size of the simulated log created in each iteration $\tilde{n} = n = 400$ and the required number of tolerable parameter sets (r) and the number of verification replications (v) are both set equal to 10.

4.2.3 Parameter Adjustment Method

A final key operationalization effort involves specifying a method to adjust Ψ across iterations. To this end, the observation that the mean of a gamma distribution μ equals $\alpha\beta$ is used [5]. The mean IAT fixes the relationship between both parameters. Hence, given the mean IAT, adjustments in one parameter automatically generates changes in the other parameter. The mean IAT is mined from the original event log by considering the time between the start timestamps of the first and last entity. Dividing the length of this time frame by the number of arrivals in this period renders an approximation of μ. Given μ, the adjustment of Ψ can be brought down to varying $\tilde{\alpha}$ and changing the value of $\tilde{\beta}$ according to the relationship $\tilde{\beta} = \mu / \tilde{\alpha}$. The adjustment of $\tilde{\alpha}$ across iterations occurs as follows:

- When $\tilde{q} > q + \delta$ in the current iteration, too many entities have been queueing in the simulated log. As a consequence, $\tilde{\alpha}$ is increased for the following iteration as this increases the mean IAT for a given scale parameter. The adjustment size is determined by applying a percentage increase to $\tilde{\alpha}$ corresponding to the percent point deviation between $\tilde{q}$ and q. However, as there is no linear relationship between $\tilde{\alpha}$ and $\tilde{q}$, this value is smoothed downward to avoid too large adjustments. More specifically, it is rounded down to the nearest negative power of 10, e.g. a calculated adjustment of 0.03 results in an actual increase in $\tilde{\alpha}$ of 10^{-2} or 0.01.
- When $\tilde{q} > q - \delta$ in the current iteration, too few entities have queued upon arrival. Consequently, $\tilde{\alpha}$ is decreased as this reduces the mean IAT for a given scale parameter. The size of the parameter decrease is determined analogously to the previous situation.
- When $q - t \leq \tilde{q} \leq q + t$, $\{\tilde{\alpha}, \tilde{\beta}\}$ is recorded in $\boldsymbol{\Phi}$. In order to explore the entire range of parameter values that lead to tolerable queue percentages, a large adjustment occurs to push $\tilde{q}$ outside the tolerance limits in the next iteration. The direction of this adjustment is determined by the value of $\tilde{q}$ for the current and two prior iterations compared to q. If $\tilde{q} > q$ in the current iteration, $\tilde{\alpha}$ is doubled for the next iteration to reduce $\tilde{q}$, unless for the two prior iterations $\tilde{q} < q$. In the latter case, $\tilde{\alpha}$ is halved to explore another parameter region. The inverse holds when $\tilde{q} < q$ in the current iteration. When $\tilde{q} = q$ for the current iteration, the three prior iterations are taken into consideration, where the third lag serves as a tie-breaker.

5 Evaluation

5.1 Experimental Design

The performance of ARPRA is evaluated using the operationalization outlined in Sect. 4.2. As the presented algorithm aims to provide an improved method to mine an IAT input model, its performance should be compared to a benchmark approach representing the state-of-the-art on the topic. Given ARPRA's central premise that queue formation cannot be ignored, the selected benchmark approach does not include the notion of queues by assuming that entities arrive at their first recorded timestamp. Hence,

a gamma distribution can directly be fitted on IATs calculated from the event log, based on the first recorded timestamp of each case.

To compare ARPRA's performance to the benchmark approach, the airline example introduced in Sect. 2.1 is used. Given this setting, values for α and β are selected to represent the real arrival process, which forms the basis to generate an event log. Solely using this event log, parameter estimates are obtained using both the benchmark approach and ARPRA. When ARPRA outperforms the benchmark technique, the former's output should correspond more closely to the real parameter values than the latter's. This experiment is repeated for 500 real IAT distribution parameters, where α is randomly drawn from a uniform distribution between 1 and 2 and β from a uniform distribution between 5.5 and 7. These boundaries are purposefully selected such that the lower bound of the distribution mean $\alpha\beta$ still leads to a steady state situation, as the utilization factor ρ is smaller than one for each activity [4].

5.2 Evaluation Results

As indicated in Sect. 5.1, ARPRA's evaluation consists of approximating real parameters of the IAT distribution using both the benchmark approach and ARPRA. The random draws for α and β from the aforementioned uniform distributions to create an event log are visualized in Fig. 4a and b. These show that the drawn values are to a large extent evenly spread and span the entire range of possible values. The queue proportion in the event log, a guiding concept for ARPRA, is represented in Fig. 4c. The mean q equals 49.17 %, with minimum and maximum values of 18.25 % and 93.00 %, respectively.

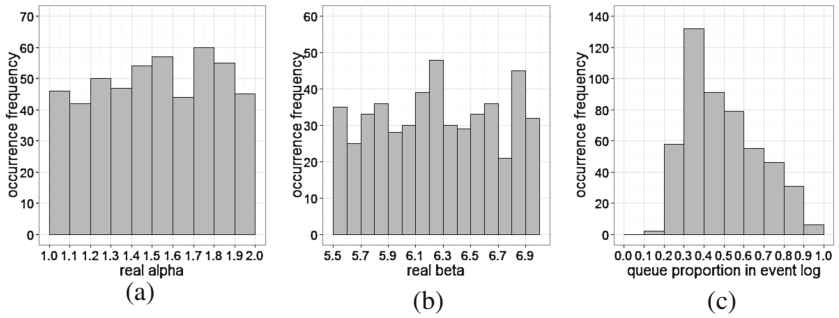

Fig. 4. Occurrence of (a) real, (b) real and (c) real queue proportion

For each of the 500 experiments, the deviation between the estimated parameters and its real value is recorded. ARPRA outperforms the benchmark approach in 498 experiments because ARPRA provides a better approximation of both the real α and β. For the remaining 2 experiments, results are mixed, i.e. one parameter is better approximated by the benchmark approach and the other one by ARPRA. As a consequence, the benchmark approach never outperforms ARPRA. Moreover, the t-test in Table 3 shows that, at a 5% significance level, ARPRA delivers an unbiased estimator for the real

parameter values, i.e. no consistent over- or underestimation is observed. In contrast, the benchmark approach renders biased estimates for both α and β.

Table 3. Results t-test on deviation between real parameters and ARPRA output

Parameter	t-value	p-value	95% confidence interval
Shape parameter α	1.07	0.2869	[–0.006; 0.021]
Scale parameter β	1.94	0.0533	[–0.001; 0.029]

Regarding the magnitude of the performance difference, key results are reported in Table 4. To put the observed deviations into perspective, Table 4 considers the percentage deviation from the real value. The results confirm that ARPRA renders more accurate approximations of the real parameters than the benchmark approach. For instance: the mean deviation from the real shape parameter equals 184.70 % for the benchmark approach and only 0.74 % for ARPRA. For the sake of completeness, the statistical significance of the performance difference is verified using a paired t-test. The null hypothesis is tested that the mean absolute value of the percentage deviation is the same for the benchmark approach and ARPRA. Table 5 shows highly significant differences and, hence, the null hypothesis can be rejected at a 5 % significance level. The benchmark approach leads to much larger deviations than ARPRA.

Table 4. Percentage deviation between real parameters and obtained estimates

Key figure	Benchmark approach	ARPRA
Shape parameter α		
Mean deviation	184.70 %	0.74 %
Quartile 1 / Quartile 3	70.66 % / 205.00 %	–8.40 % / 8.66 %
Standard deviation	211.47 % points	15.51 % points
Scale parameter β		
Mean deviation	–54.79 %	1.42 %
Quartile 1 / Quartile 3	–66.90 % / –41.10 %	–8.46 % / 8.73 %
Standard deviation	17.17 % points	16.39 % points

Table 5. Paired t-test of absolute value of percentage deviation from real α and β

Parameter	t-value	p-value	95% confidence interval
Shape parameter α	28.69	$< 2.2 \cdot 10^{-16}$	[155.08; 191.51]
Scale parameter β	57.31	$< 2.2 \cdot 10^{-16}$	[41.36; 44.30]

It can be concluded that ARPRA presents an important improvement over the benchmark approach. Consequently, it is shown that queue formation has to be taken into account when mining an IAT input model. Given the implications of inaccurate IAT

input models, illustrated in Sect. 2.1, BPS model construction can benefit from ARPRA when an IAT input model needs to be mined from an event log.

6 Conclusion

This paper focused on process mining support for IAT modelling when constructing a simulation model. The main contribution of this work is twofold. Firstly, an IAT input model taxonomy for process mining is developed, showing that the current approach in literature is only appropriate when no queues are formed for the start activity. Secondly, ARPRA is introduced, which is the first to explicitly take the notion of queues into account when mining an IAT input model. When the algorithm is operationalized for gamma distributed IATs, the conducted experiments show that: (i) ARPRA provides an unbiased estimator for both distribution parameters and (ii) ARPRA significantly outperforms a benchmark approach ignoring queue formation.

Future work will focus on the development of a more advanced parameter search strategy, taking into account the non-linear relationship between the distribution parameters and the queue proportion. Moreover, a sensitivity analysis will be performed to investigate ARPRA's sensitivity to the queue proportion in the original log, the size of the original log, etc. Finally, ARPRA can be extended to mine an IAT input model when (i) no a priori distribution is assumed, (ii) input model distributions and/or parameters vary over time, (iii) only start or end events are recorded and (iv) more complex resource behavior such as batch processing is present in the process under consideration.

References

1. de Smet, L.: Queue mining: combining process mining and queuing analysis to understand bottlenecks, to predict delays, and suggest process improvements. Master thesis, Eindhoven University of Technology (2014)
2. Kelton, W.D., Sadowski, R.P., Zupick, N.B.: Simulation with Arena. McGraw-Hill, New York (2015)
3. Henderson, S.G.: Input modeling uncertainty: why do we care and what should we do about it. In: Proceedings of the 2003 Winter Simulation Conference, pp. 90–100 (2003)
4. Hillier, F.S., Lieberman, G.J.: Introduction to Operations Research. McGraw-Hill, New York (2010)
5. Law, A.M.: Simulation Modeling and Analysis. McGraw-Hill, New York (2007)
6. Martin, N., Depaire, B., Caris, A.: The use of process mining in a business process simulation context: overview and challenges. In: Proceedings of the 2014 IEEE Symposium on Computational Intelligence and Data Mining, pp. 381–388 (2014)
7. Melão, N., Pidd, M.: Use of business process simulation: a survey of practitioners. J. Oper. Res. Soc. **54**(1), 2–10 (2003)
8. Robinson, S.: Simulation: the Practice of Model Development and Use. Wiley, Chichester (2004)
9. Rozinat, A., Mans, R.S., Song, M., van der Aalst, W.M.P.: Discovering simulation models. Inform. Syst. **34**(3), 305–327 (2009)

10. Senderovich, A., Weidlich, M., Gal, A., Mandelbaum, A.: Queue mining – predicting delays in service processes. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 42–57. Springer, Heidelberg (2014)
11. Song, M., van der Aalst, W.M.P.: Supporting process mining by showing events at a glance. In: Proceedings of the 17th Annual Workshop on Information Technologies and Systems, pp. 139–145 (2007)
12. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
13. van der Aalst, W.M.P.: Business process simulation survival guide. BPM Center Reports no. BPM-13-11 (2013)
14. van der Aalst, W.M.P., van Hee, K.M., ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets: classification, desirability and analysis. Form. Asp. Comput. **23**, 333–363 (2011)