Queue Mining – Predicting Delays in Service Processes

Arik Senderovich[1], Matthias Weidlich[2], Avigdor Gal[1], and Avishai Mandelbaum[1]

[1] Technion – Israel Institute of Technology
{sariks@tx,avigal@ie,avim@ie}.technion.ac.il
[2] Imperial College London
m.weidlich@imperial.ac.uk

Abstract. Information systems have been widely adopted to support service processes in various domains, *e.g.*, in the telecommunication, finance, and health sectors. Recently, work on process mining showed how management of these processes, and engineering of supporting systems, can be guided by models extracted from the event logs that are recorded during process operation. In this work, we establish a queueing perspective in operational process mining. We propose to consider queues as first-class citizens and use queueing theory as a basis for queue mining techniques. To demonstrate the value of queue mining, we revisit the specific operational problem of *online delay prediction*: using event data, we show that queue mining yields accurate online predictions of case delay.

1 Introduction

The conduct of service processes, *e.g.*, in the telecommunication and health sectors, is heavily supported by information systems. To manage such processes and improve the operation of supporting systems, event logs recorded during process operation constitute a valuable source of information. Recently, this opportunity was widely exploited in the rapidly growing research field of process mining. It started with mining techniques that focused mainly on the control-flow perspective, namely extracting control-flow models from event logs [1] for qualitative analyses, such as model-based verification [2].

In recent years, research in process mining has shifted the spotlight from qualitative analysis to (quantitative) *online* operational support ([3], Ch. 9). To provide operational support, the control-flow perspective alone does not suffice and, therefore, new perspectives are mined. For example, the time perspective exploits event timestamps and frequencies to locate bottlenecks and predict execution times.

To date, operational process mining is largely limited to black-box analysis. That is, observations obtained for single instances (cases) of a process are aggregated to derive predictors for the behaviour of cases in the future. This approach can be seen as a regression analysis over individual cases, assuming that they are executed largely independently of each other. In many processes, however, cases do not run in isolation but *multiple cases* compete over scarce resources. Only some cases get served at a certain point in time (complete execution of an activity and progress in process execution) while others must wait for resources to become available. Cases that did not get served are enqueued and consequently delayed.

In this work, we establish a queueing perspective in process mining. We propose to consider queues as first-class citizens and refer to the various tasks of process mining that involve the queueing perspective as *queue mining*. Further, we present techniques for queue mining following two different strategies. First, following the traditional approach of operational process mining, we enhance an existing technique for time prediction [4] to consider queues and system load. Second, we argue for the application of queueing theory [5, 6] as a basis to model, analyse, and improve service processes. Specifically, we mine delay predictors that are based on well-established queueing models and results.

To demonstrate the value of queue mining, we address the specific operational problem of *online delay prediction*, which refers to the time that the execution of an activity for a particular case is delayed due to queueing effects. In addition to the definition of mining techniques for different types of predictors, we also present a comprehensive experimental evaluation using real-world logs. It shows that the proposed techniques improves prediction and that predictors grounded in queueing theory have, in most cases, superior prediction accuracy over regression-based time prediction.
The contributions of our paper can be summarized as follows:

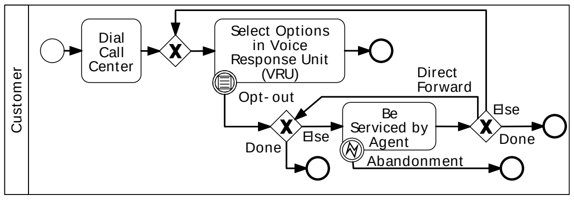
- The queueing perspective is introduced as a novel agenda in process mining, and queue mining is positioned as a new class of mining tasks.
- For the online delay prediction problem, we present techniques for the derivation of predictors of two types; those that enhance existing regression-based techniques for process mining and those that are grounded in queueing theory.
- The value of queue mining techniques is demonstrated empirically by their accuracy in delay prediction. Also, this paper contributes to queueing theory by validating, against real data, delay predictors that have been so far tested only on synthetic simulation runs.

The remainder of this paper is organized as follows. The next section provides motivation for the queueing perspective and background on queueing models. Section 3 defines the queue log as a basis to our queue mining methods and Section 4 states the delay prediction problem. Section 5 introduces delay predictors and brings in mining methods for these predictors. Section 6 presents our experiments and discusses their results. We review related work in Section 7 and conclude in Section 8.

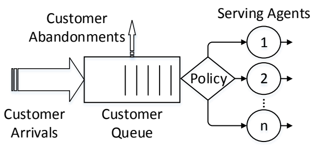
2 Background

We illustrate the need for a queueing perspective, in operational models for services processes, with the example of a bank's call centre. Figure 1(a) depicts a BPMN [7] model of such a process, which focuses on the control-flow of a case, that is, a single customer. The customer dials in and is then connected to a voice response unit (VRU). The customer either completes service within the VRU or chooses to opt-out, continuing to an agent. Once customers have been served by an agent, they either hang-up or, in rare cases, choose to continue for another service (VRU or forwarding to an agent).

Although this model provides a reasonable abstraction of the process from the perspective of a single customer, it fails short of capturing important operational details. Customers that seek a service are served by one of the available agents or wait in a queue. Hence, activity 'Be Serviced by Agent' comprises a waiting phase and an actual service phase. Customers that wait for service may also abandon from the queue due to



(a) Control flow perspective



(b) Queueing perspective

Fig. 1. Example process in a call centre

impatience. To provide operational analysis for this service process and predict delay of processing, such queues and abandonments must be taken into account explicitly.

For the above example, only activity 'Be Serviced by Agent' involves significant queueing since the other activities do not rely on scarce resources of the service provider. Adopting a queueing perspective for this activity, Fig. 1(b) outlines how the activity is conducted under multiple cases arriving at the system and, thus, emphasizes that execution time of one case depend on cases that are already in the system.

The model shown in Fig. 1(b) can be viewed as a single-station queueing system, where one station is served by n homogeneous agents. Such a queueing system is described by a series of characteristics, denoted using Kendall's notation as $\mathcal{A}/\mathcal{B}/\mathcal{C}/\mathcal{Y}/\mathcal{Z}$ [8]. The arrival process ($\mathcal{A}$) is defined by the joint distribution of the inter-arrival times. Whenever no assumption regarding the arrival process is made, $\mathcal{A}$ is replaced by G for general distribution. The processing duration of a single case ($\mathcal{B}$) is described by the distribution of service time. The total number of agents at the queueing station is denoted by $\mathcal{C}$, which stands for capacity. When a case arrives and all agents are busy, the new arrival is queued. The maximum size of the system, $\mathcal{Y}$, can be finite, so that new customers are blocked if the number of running cases is larger than $\mathcal{Y}$. In call centres, which provides our present motivation, $\mathcal{Y}$ is practically infinite and can be omitted. Once an agent becomes available and the queue is not empty, a customer is selected according to a routing policy $\mathcal{Z}$. The most common policy is the FCFS (First Come First Served) policy and in such cases $\mathcal{Z}$ is also omitted.

Queueing models may include information on the distribution of customer (im)patience ($\mathcal{G}$), added following a '$+$' sign at the end of Kendall's notation. For mathematical tractability and sometimes backed up by practice, it is often assumed that sequences of inter-arrival times, service times and customer (im)patience are independent of each other, and each consists of independent elements that are exponentially distributed. Then, $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{G}$ are replaced by Ms, which stands for Markovian. For example, a $G/M/n + M$ model assumes that arrivals come from a general distribution, service times are exponential, agent capacity is of size n, queue size is infinite, routing policy is FCFS and (im)patience is exponentially distributed.

3 The Queue Log

To mine the queueing perspective of a service process, events that have been recorded during operation must be available. Most existing techniques for process mining assume that these events are process related, indicating start and end of activity execution, cf. [3]. For the queueing perspective we need to record events that relate to queueing transactions, including queue entrance, abandonment, and service start and end. Below, we define a Q-Log for a single queueing station representing a single activity of the service process, but note that the definition can be easily extended to more general queueing models (*e.g.*, multi-class customers and queueing networks).

Definition 1 (Single-Station Q-Log) *A* single-station queue log Q *is a finite sequence* $Q : \mathbb{N}^{+} \rightarrow \mathcal{Q}$ *over queue events* $(t, c, p, a) \in \mathcal{Q}$*, where*

– $t \in \mathbb{N}^{+}$ *is a timestamp,*
– $c \in \mathbb{N}^{+}$ *is a unique case (customer) identifier,*
– $p \in \mathcal{P} = \{qEntry, qAbandon, sStart, sEnd\}$ *is a state change for the case,*
– $a \in \mathbb{N}^{+}$ *is a unique agent identifier, set only when* $p \in \{sStart, sEnd\}$.

Below, we write (t_i, c_i, p_i, a_i) for the i-th queue event $Q(i)$ and $|Q|$ for the length of the log. Also, to keep the notation concise, we assume that a Q-Log is ordered by the timestamp, i.e., $t_i \leq t_j$ for all $1 \leq i < j \leq |Q|$.

We refer to *queue mining* as a class of techniques that take a Q-Log as input and derive a model for the operational analysis of the queue and service.

4 The Delay Prediction Problem

In this section, we elaborate on the problem of *online delay prediction*, which will be solved using queue mining in this work. The phenomena of delay has been a popular research subject in queueing theory, see [9]. The interest in delay prediction is motivated by psychological insights on the negative effect of waiting on customer satisfaction [10]. Field studies have found that seeing the line ahead moving and getting initial information on the expected delay, both have a positive effect on the waiting experience [11, 12]. Thus, announcing the expected delay in an online fashion improves customer satisfaction.

We refer to the customer, whose delay time we wish to predict as the *target-customer*. The target-customer is assumed to have infinite patience, i.e. the target customer will wait patiently for service, without abandoning, otherwise our prediction becomes useless. However, the influence of abandonments of other customers on the delay time of the target-customer is taken into account. In some service processes (e.g., the one introduced in Section 2), customers may return to a queue after they have been served. In the remainder, we focus on predicting the first delay. Many real-world applications treat returning customers as a negligible phenomenon. For instance, returning customers are only 3% of the cases in the real-life process that we consider in Section 6. We formalize the delay prediction problem as follows.

Problem 1 (Online delay prediction) *Let* W *be a random variable that measures the first delay time of a target-customer. Denote by* $\widehat{W}$ *the predictor for* W*. Then, the* online delay prediction *problem aims at identifying an accurate and precise predictor* $\widehat{W}$.

Accuracy and precision are common measures for prediction (see, for example, the use of precision and recall in information retrieval [13]). In our experiments (Section 6), we use, as concrete measures, absolute bias for accuracy and root mean squared error (RMSE) for precision.

5 Delay Predictors: Queue Mining in Action

In order to solve the delay prediction problem, we propose mining techniques for three classes of delay predictors that implement two different strategies. First, we follow the traditional regression-based analysis of operational process mining and enhance a technique presented by van der Aalst et al. for time prediction [4]. We do so by considering a queue to be yet another activity and including a notion of system load in the mining. Our second and third class of predictors, in turn, follow a completely different strategy and are grounded in queueing theory. Here, we present mining techniques for delay predictors that arise directly from a queueing model. These yield delay predictors that are based on queueing theory and a congestion law (*the snapshot principle*).

5.1 State Transition System Predictors

Idea Our first predictor follows the idea of integrating queueing information into techniques for operational process mining by considering queues as separate activities of the process. Following this line, we rely on a state-of-the-art approach for time prediction based on transition systems that has been presented by van der Aalst et al. [4]. In essence, this approach mines a transition system from a log of recorded activity executions. To this end, different abstractions may be applied to states and state transitions as they are recorded in the log. For instance, a state in the transition system may be defined by the sequence of all activities executed in a case so far, the set of these activities, or only the last executed activity. This abstraction of the process behaviour is then annotated with timing information, *e.g.*, the time needed to finish the process from a particular state averaged over all respective cases.

To use this approach for delay prediction, the queueing phase and the service phase of an activity in a service process are treated as two separate steps and a state is defined as the sequence of executed activities. Then, when a customer is enqueued, we enter an initial state ⟨queue⟩ in the transition system. When a customer enters service, we move to a state ⟨queue, service⟩ and, after service completion, to a state ⟨queue, service, end⟩. If a customer abandons the queue, we transition to a state ⟨queue, end⟩ instead.

The transition system constructed with this approach is based on an evaluation of all cases in isolation and, thus, neglects the influence of other cases. To account for this aspect, we extend the state abstraction to include system load as a context-factor in the spirit of [25]; we represent system load by the number of customers in queue $L(t)$. In practice, $L(t)$ may vary greatly over time, so that including its absolute values leads to a ‘state explosion’. Therefore, we partition $L(t)$ into k clusters, and consider the ⟨queue⟩ state combined only with these k clusters. For $k = 3$, i.e., assuming that the system can be in one of three load states, namely ‘high’, ‘moderate’, and ‘typical’, the resulting transition system is illustrated in Fig. 2.

Queue Mining Given a Q-Log, construction of the first version of the transition system is straight-forward. For the resulting system, state ⟨queue⟩ is annotated with past delay times for customers who waited and eventually got served. Then, the average over these values yields a predictor $\widehat{W}_{PTS}$, referred to as plain transition system predictor.

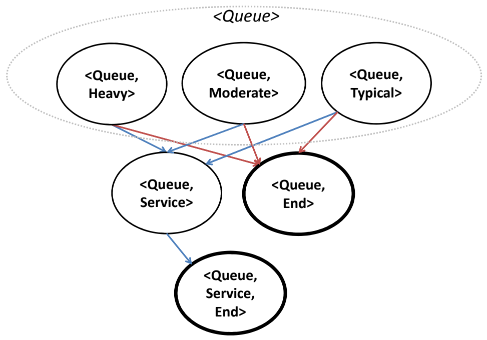


Fig. 2. Transition system extended for three load-classes - High, Moderate, and Typical

For the second predictor, we first derive the load clusters using k-means clustering [14]. Based on the partition of $L(t)$ into k load types, we derive the respective transition system. Lastly, we annotate each of the states that represent waiting in a queue under a certain system load with previous delay times. For each state, we receive a different delay predictor, winding up with k predictors $\widehat{W}_{KTS}$, referred to as k-loads transition system predictors. From a statistical viewpoint, the k-loads transition system method may be considered as a regression model that uses system load to predict delays.

5.2 Queueing Model Predictors

Idea Our second class of predictors does not follow a regression analysis, but is grounded in queueing theory. These predictors relate to the $G/M/n+M$ model, so that upon the arrival of a target-customer, there are n homogeneous working agents at the station. We denote the mean service time by m and assume that service duration is exponentially distributed. Therefore, the service rate of an individual agent is $\mu = 1/m$. Impatient customers may leave the queue and customer individual patience is exponentially distributed with mean $1/\theta$, i.e., the individual abandonment rate is θ. Whenever customers do not abandon the system ($\theta = 0$), the model reduces to $G/M/n$.

We define two delay predictors based on the $G/M/n$ and the $G/M/n+M$ models, respectively. We refer to the first predictor as queue-length (based) predictor (QLP) and to the second as queue-length (based) Markovian (abandonments) Predictor (QLMP) [15]. As their names imply, these predictors use the queue length (in front of the target customer) to predict its expected delay. We define the queue-length, $L(t)$, to be a random variable that quantifies the number of cases that are delayed at time t. The QLP for a target customer arriving at time t is:

$$\widehat{W}_{QLP}(L(t)) = \frac{(L(t)+1)}{n\mu} \tag{1}$$

with n being the number of agents and μ being the service rate of an individual agent.

The QLMP predictor assumes finite patience and is defined as:

$$\widehat{W}_{QLMP}(L(t)) = \sum_{i=0}^{L(t)} \frac{1}{n\mu + i\theta}. \tag{2}$$

Intuitively, when a target-customer arrives, it may progress in queue only if customers that are ahead of him enter service (when an agent becomes available, at rate $n\mu$) or abandon (at rate $i\theta$ with i being the number of customers in queue). For the QLP, $\theta = 0$ and thus the QLMP predictor (Eq. 2) reduces to the QLP predictor (1).

Queue Mining Given a Q-Log Q that is up-to-date at time t, we extract the current number of customers in queue and the current number of working agents n as follows (note that $Q(i) = (t_i, c_i, p_i, a_i)$ denotes the i-th queue event in Q):

– $L(t) = |\{(t_i, c_i, p_i, a_i) \mid p_i = qEntry \wedge \forall\, (t_j, c_j, p_j, a_j), i \leq j : c_j \neq c_i\}|$,
– $n = |\{(t_i, c_i, p_i, a_i) \mid p_i = sStart \wedge \forall\, (t_j, c_j, p_j, a_j), i \leq j : a_j \neq a_i\}|$.

To obtain QLP and QLMP, we only need to estimate two parameters, namely the service rate (μ) and the abandonment rate (θ). To estimate μ, we go over all pairs of queue events $(t_i, c_i, p_i, a_i), (t_j, c_j, p_j, a_j)$ for which $c_i = c_j$, $p_i = sStart$ and $p_j = sEnd$, and average over $t_j - t_i$, which is the service time for case c_i. The result is an unbiased estimator $\hat{m}$ for the mean service time. We can then deduce a naive moment estimator for $\hat{\mu}$, $\hat{\mu} = 1/\hat{m}$ [16].

Estimation of θ is based on a statistical result that relates it to the total number of abandonments and the total delay time for both served and abandoned customers, cf., [17]. Practically, we obtain the number of abandonments by counting the respective queue events. The total delay time is derived by summarizing delay durations for all customers that experienced queueing. Then, the abandonment rate is estimated as:

$$\hat{\theta} = \frac{|\{(t_i, c_i, p_i, a_i) \mid p_i = qAbandon\}|}{\sum_{(t_i,c_i,p_i,a_i),(t_j,c_j,p_j,a_j),c_i=c_j,p_i=qEntry,p_j\in\{qAbandon,sStart\}} (t_j - t_i)}. \tag{3}$$

5.3 Snapshot Predictors

Idea An important result in queueing theory is the *(heavy-traffic) snapshot principle* (see [18], p. 187). A heavy-traffic approximation refers to the behaviour of a queue model under limits of its parameters, as the workload converges to capacity. In the context of Problem 1, the snapshot principle implies that under the heavy-traffic approximation, delay times (of other customers) tend to change negligibly during the waiting time of a single customer [15]. We define two snapshot predictors: Last-customer-to-Enter-Service (LES or $\widehat{W}_{LES}$) and Head-Of-Line (HOL or $\widehat{W}_{HOL}$). The LES predictor is the delay of the most recent service entrant, while the HOL is the delay of the first customer in line.

In real-life settings, the heavy-traffic approximation is not always plausible and thus the applicability of the snapshot principle predictors should be tested ad-hoc, when working with real data sets. Results of synthetic simulation runs, conducted in [15], show that the LES and HOL are indeed appropriate for predicting delays.

Queue Mining Given a Q-Log Q that is up-to-date at time t, we mine these predictors as follows. For the LES, the case c_i that last entered service is the one that satisfies the following condition: there is a queue event (t_i, c_i, p_i, a_i) with $p_i = sStart$ and for all $(t_j, c_j, p_j, a_j), i \leq j$ it holds $p_j \neq sStart$. Then, with $(t_q, c_q, p_q, a_q), c_q = c_i, p_q = qEntry$ as the event indicating queue entrance of the respective customer, the predictor is derived as $\widehat{W}_{LES} = t_i - t_q$.

For the HOL, we observe the case c_i that is still in queue, but would be the next to enter service, i.e., there is a queue event (t_i, c_i, p_i, a_i) with $p_i = qEntry$ and for all $(t_j, c_j, p_j, a_j), j \leq i$ with $p_j = qEntry$ there exists another queue event $(t_k, c_k, p_k, a_k), j \leq k$ with $c_j = c_k$ and $p_k \in \{qAbandon, sStart\}$. Then, the predictor is derived as $\widehat{W}_{HOL} = t - t_i$. Note that this technique for mining the HOL holds only if the FCFS policy is assumed, otherwise the order of $qEntry$ timestamps does not impose the same order on $sStart$.

6 Evaluation

To test the various delay predictors we ran a set of experiments on a real-life queue log. Our experiments show that the snapshot predictors outperform other predictors, in virtually every experimental setting considered. For the predictors based on transition systems we observe that the plain predictor performed poorly, whereas the one integrating the system load leads to significant improvements. Both queueing model-based predictors, in turn, performed worse than the snapshot predictors, since the queueing model assumptions are often violated in real-life processes.

Below, we first describe the real-life queue log used for our experiments (Section 6.1). Then, we define the evaluation's performance measures (Section 6.2), describe our experimental setup (Section 6.3) and report on the main results (Section 6.4). We conclude with a discussion (Section 6.5).

6.1 Data Description

The experiments were conducted on a real-life queue log of a call center of an Israeli bank. The data comes from the Technion laboratory for Service Enterprise Engineering (SEELab)[3]. The dataset contains a detailed description of all operational transactions that occurred within the call center, between January 24th, 2010 and March 31st, 2011. The log contains, for an average weekday, data on approximately 7000 calls.

For our delay prediction experiments, we selected three months of data: January 2011-March 2011 (a queue log of 879591 records). This amount of data enables us to gain useful insights into the prediction problem, while easing the computational complexity (as opposed to analysing the entire data set). The three months were selected since they are free of Israeli holidays. In our experiments, we focused only on cases that demanded 'general banking' services, which is the majority of calls arriving into the call center (89%). This case selection is appropriate, since our queueing models assume that customers are homogeneous.

[3] http://ie.technion.ac.il/Labs/Serveng

We divided the experimental queue log into two subsets: a training log and a test log. This is common practice when performing statistical model assessment [14]. The training log comprises all calls that arrived between January 1st, 2011 and February 28th, 2011; a total of 250488 delays and 247281 completed services. The test log consisted of delays that occurred during March 2011; a total of 117709 delays. We addressed each delayed customer in the test log as the target-customer for whom we aim at solving Problem 1.

6.2 Performance Measures

To evaluate the quality of the delay predictors, we introduce two performance measures: *absolute bias*, for accuracy, and *root mean squared error* (RMSE), for precision. The absolute bias is defined as:

$$|Bias(\widehat{W})| = |E[\widehat{W}] - W|, \tag{4}$$

with W being the delay and $\widehat{W}$ being the delay predictor. We define a point estimate for the absolute bias as:

$$|\widehat{Bias}| = |\frac{1}{k}\sum_{i=1}^{k}(d_i - p_i)|, \tag{5}$$

with $i = 1, ..., k$ being the i-th test-log delay out of k delays, d_i the real duration of the i-th delay and p_i the corresponding predicted delay; $|Bias(\widehat{W})| > 0$ indicates a systematic error in our predictor, thus low accuracy.

For precision define RMSE as:

$$RMSE(\widehat{W}) = \sqrt{E[(W - \widehat{W})^2]}. \tag{6}$$

We consider a point estimate for the RMSE to be the *root average squared error* (RASE), namely,

$$RASE = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(d_i - p_i)^2}, \tag{7}$$

with d_i and p_i defined as before. Low RMSE indicates that the corresponding predictor is precise.

We consider the RMSE (and precision) to be more significant, penalizing for any type of deviation from the real delay. In contrast, the absolute bias may result in 0 (high accuracy), but deviate strongly from the delay predictor (*e.g.*, deviating strongly both above and below the real delay). We thus use accuracy as a 'compass' to detect systematic errors in model assumptions, but consider precision to be the indicator for quality of prediction.

6.3 Experimental Setup

The controlled variable in our experiments is the *prediction method* (or the delay predictor). Six various methods are used according to the predictors defined in Section 5,

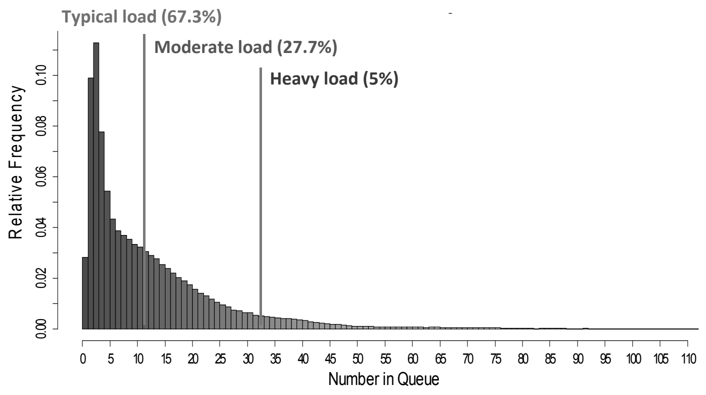


Fig. 3. Relative frequency histogram for number of customers in queue, from training log

namely the QLP, QLMP, LES, HOL, PTS and KTS predictors. The uncontrolled variables are the two performance measures $|\widehat{Bias}|$ and RASE.

As a preliminary step, we mined the K-loads transition system (applied the KTS) from the training log with $K = 3$. The result was a clustering of the queue-length ($L(t)$) into 3 classes: 'heavy load', 'moderate load,' and 'typical load'. Figure 3 shows the relative frequency histogram of $L(t)$. The vertical lines depict the partition that resulted from running the 3-means algorithm on the training log. For example, 'typical load' is the region where $0 \leq L(t) \leq 12$. Given the partition to $K = 3$ classes of load, we tested our predictors on four different experimental scenarios. Scenario I considered the entire test log and thus we refer to it as the 'all loads' scenario, while Scenarios II-IV relate to the three load-clusters and to delays that are associated with these clusters.

Table 1. Test delays statistics

Load	Range $L(t)$	% of test delays	Average Delay (sec)	Standard Deviation (sec)
Scenario I	0-∞	100.0%	71.5	84
Scenario II	32-∞	5.0%	252.0	125
Scenario III	$12-32$	27.7%	129.0	74
Scenario IV	$0-12$	67.3%	34.0	44

Table 1 summarizes the statistics for the four scenarios. Range $L(t)$ corresponds to the vertical lines in Figure 3, *e.g.*, for typical load the Range $L(t)$ is between 0 and 12. Due to the increase in standard deviation of the delay times as the load increases, we expect that the error (in seconds) will be smaller for the typical load and increasingly larger for the two other load types.

6.4 Results

Figure 4 presents the absolute bias for all six predictors under the four load scenarios.

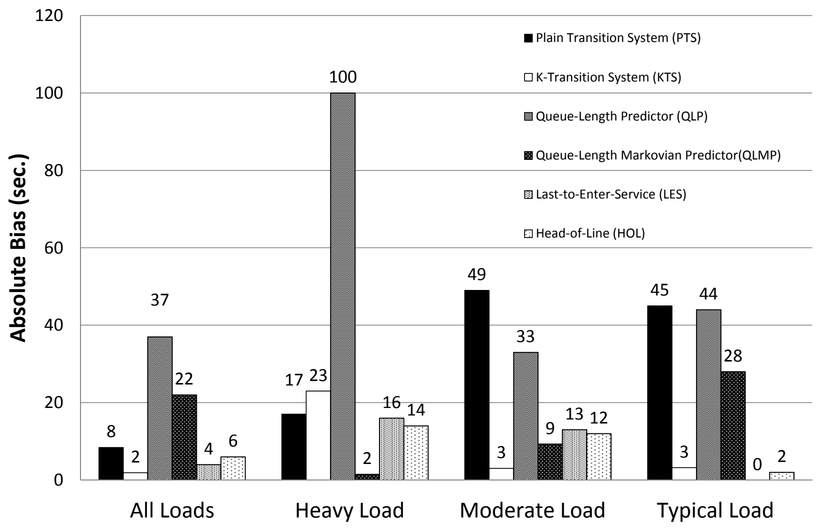


Fig. 4. Sampled bias - Test set delays

The PTS predictor presents a near-zero bias in Scenario I, but when observing its bias across scenarios we note a much larger bias. This originates in the insensitivity of the PTS predictor to system load. The KTS has a negligible bias in all scenarios, except for the one representing heavy load. This result hints at the existence of a finer partitioning of the heavy load scenario.

For the queue-length based predictors (QLP and QLMP), we observe that both predictors appear to be biased. This may point towards violations in the queueing model assumptions. The bias of the snapshot predictors (LES and HOL) is small across scenarios, indicating an absence of a systematic error in these predictors. This observation supports the applicability of the snapshot predictors to service processes in call centers.

Figure 5 presents the RASE in seconds. Snapshot predictors are superior across all scenarios, improving over the PTS by 34%-46%. Note that both snapshot predictors perform identically in terms of RASE. This empirical proximity between the LES and the HOL, under certain assumptions, has theoretical background in [15] (Theorem 4.4).

The QLP performed worse than PTS across scenarios, except the moderate load scenario, while the QLMP outperformed the PTS in all scenarios except the typical load scenario. In the moderate load scenario, the QLMP performs almost as well as the snapshot predictors. The KTS outperforms both the PTS and queueing model predictors, in all scenarios, except the moderate load scenario.

6.5 Discussion

In the remainder of this section, we divide our discussion according to the three predictor classes.

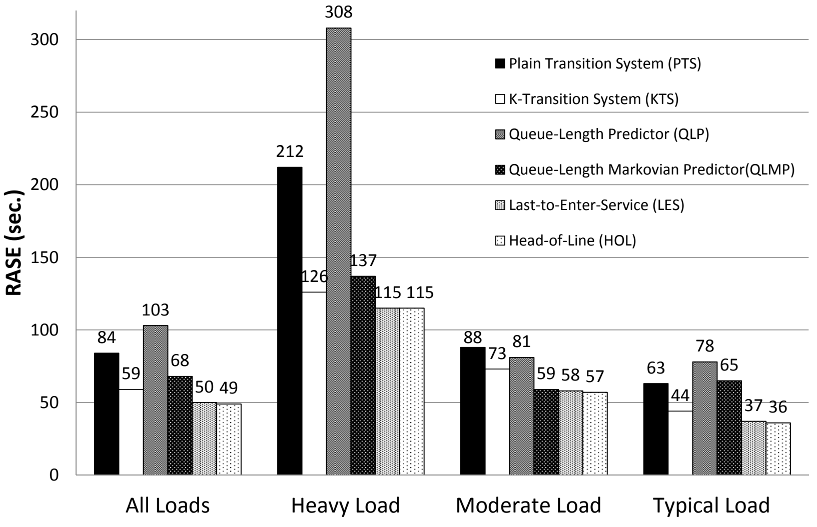


Fig. 5. Root-average squared error in seconds

Transition System Methods: There is no (single) Steady-State
In both transition system methods we consider past delays of customers with similar path-history, when predicting the delay of the target-customer. The problem with the PTS is that, when applied to our real-life process, it considers all past delays. Considering all past delays is appropriate in steady-state analysis, *i.e.*, when the relation between demand and capacity does not vary greatly over time. The KTS (with $K = 3$) performs significantly better, since it captures three different steady-states (three system loads). The performance of this method is second best, in most scenarios, only to snapshot predictors, which may in turn imply that the existence of three steady-states is indeed a plausible hypothesis.

Queue-Length Predictors: Model Assumptions Matter
The queueing model predictors consider the time-varying behaviour of the system and attempt to quantify the system-state based on the number of delayed cases. The QLP fails in *accuracy and precision* for most scenarios, since it assumes that customers have infinite patience, which is seldom the case in call center processes. We presume that the QLP would perform better for processes with negligible abandonment rates.

On the other hand, the QLMP outperforms the PTS for most scenarios both in *accuracy and precision*. Therefore, accounting for customer (im)patience is indeed relevant in the context of call centers, and other processes in which abandonments occur [20]. In contrast, the QLMP is inferior when compared to snapshot predictors or the KTS predictor. This phenomena can be explained by deviations between model assumptions and reality. We demonstrate one possible deviation by conducting a short (descriptive) statistical analysis that is relevant for both the QLP and the QLMP. Figure 6 presents the mean service time over a single day (January 2nd, 2011, which is a typical Sunday in our training log).

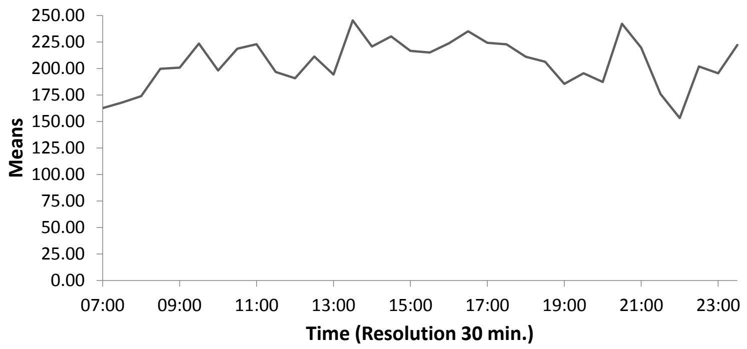


Fig. 6. Mean service time during a typical Sunday (January 2nd, 2011)

The horizontal axis presents the time-of-day in a 30 minutes resolution and the vertical axis presents the mean service time in seconds, during each of the 30 minutes. We see that the mean service time is mostly stable, but nonetheless violations do occur during several time points. This fluctuation over the day may cause deterioration in overall performance of both QLP and QLMP, since these predictors assume constant mean service time. We have shown similar violations for both the constant (im)patience time and the exponential service times assumptions, but we do not present them in this paper, due to space considerations.

Snapshot Principle Predictors: Recent History Dominates in Time-Varying Systems
Throughout our experiments, snapshot predictors have shown the largest improvement over the PTS method and outperformed the rest in terms of precision. Thus, we conclude that for the considered queueing process (of a call center), an adequate delay prediction for a newly enqueued customer would be the delay of either the current head-of-the-line (HOL) or the delay of the last-customer-to-enter-service (LES). Our main insight is that in time-varying systems, such as call center, one must consider only recent delay history when inferring on newly arriving cases.

The snapshot principle was shown to work well with multiple service stations as well [18, 19]. Therefore, investigating the use of this principle to a queueing network with a complex underlying process may provide competent prediction.

7 Related Work

Our work is related to two streams of research, namely operational process mining and delay prediction in queueing theory.

Lately, process mining has seen a remarkable uptake, providing tools for the creation of process models from event data, over the assessment of the conformance of models and events, to extensions of models for operational support, see [3] for a recent overview. Despite the wide-spread focus on the control flow perspective, process mining techniques would benefit from additional information, such as time and resource utilization. In particular, several approaches addressed the problem of predicting process completion times for running cases. Van der Aalst et al. [21] highlight the importance of capturing resource utilization appropriately and provide techniques for mining a simulation model. The approach creates a Coloured Petri net that comprises resource and timing information and serves as the basis for time prediction. Rogge-Solti and Weske [22] follow a similar approach, but ground the analysis in a probabilistic model, formalised as a stochastic Petri net. Then, Monte-Carlo simulation allows for predicting completion time. A generic framework for time prediction based on state transition systems constructed from process logs was developed in [4]. Our work complements these techniques by focusing on the delay of a case when executing a certain activity instead of estimating the overall completion time. To this end, we add the queueing perspective to process mining and introduce techniques for queue mining. These techniques yield prediction models that take the notion of a queue into account explicitly. We relied on queue models and also enhanced the method based on transition systems of [4] to take queueing effects into account. However, our evaluation illustrated that the predictors that are derived by queue mining and grounded in queue models show superior performance.

Predicting queueing delays has been a popular research subject in queueing theory; see [9] for an overview. Statistical techniques for estimating delays were applied mainly for systems in steady-state [23, 17]. Real-time delay predictors that do not assume steady-state, in analogy to Problem 1 addressed in this work, were proposed in [24, 15]. We use these predictors as a basis to our queue mining techniques and address the derivation of these predictors from event data.

8 Conclusion

In this paper, we showed how to consider a queueing perspective in operational process mining for service processes. In particular, we state the problem of *online delay prediction* and provide different techniques, jointly referred to as *queue mining*, that take recorded event data as input and derive predictors for the delay of a case cause by queueing. First, we consider mining of regression-based predictors that are based on state transition systems, for which queueing information has been integrated. We further argued for predictors that are grounded in queueing theory and presented mining techniques for predictors that emerge from a queueing model, either based on queueing theory or the snapshot principle. For all predictors, we tested accuracy using a real-life queue log. Our experiments show that predictors grounded in queueing models, in most cases, provide superior performance, improving accuracy by 30%-40% on average compared to the plain regression-based method.

In future work, we intend to expand queue mining to Q-Logs that stem from complex service processes with several stations, *i.e.*, activities that involve queueing. The natural models, when considering such processes, are *queueing networks*. These models are often mathematically intractable and thus the analysis of queueing networks resorts to simulation or approximation methods in the spirit of the *snapshot principle*.

References

1. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Trans. Knowl. Data Eng. **16**(9) (2004) 1128–1142

2. van der Aalst, W.M.: Workflow verification: Finding control-flow errors using petri-net-based techniques. In: Business Process Management. Springer (2000) 161–183
3. van der Aalst, W.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
4. van der Aalst, W.M., Schonenberg, M., Song, M.: Time prediction based on process mining. Information Systems **36**(2) (2011) 450–475
5. Hall, R.W.: Queueing Methods: For Services and Manufacturing. Prentice Hall, Englewood Cliffs NJ (1991)
6. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: Queueing networks and Markov chains - modeling and performance evaluation with computer science applications. Wiley (2006)
7. Object Management Group (OMG) : Business Process Model and Notation (BPMN). (2011)
8. Kendall, D.G.: Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain. The Annals of Mathematical Statistics **24**(3) (1953) pp. 338–354
9. Nakibly, E.: Predicting waiting times in telephone service systems. Master's thesis, Technion–Israel Institute of Technology (2002)
10. Houston, M.B., Bettencourt, L.A., Wenger, S.: The relationship between waiting in a service queue and evaluations of service quality: A field theory perspective. Psychology and Marketing **15**(8) (1998) 735–753
11. Carmon, Z., Kahneman, D.: The experienced utility of queuing: real time affect and retrospective evaluations of simulated queues. Technical report, Duke University (1996)
12. Larson, R.C.: Perspectives on queues: Social justice and the psychology of queueing. Operations Research **35**(6) (1987) 895–905
13. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
14. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics. Springer New York Inc., New York, NY, USA (2001)
15. Ibrahim, R., Whitt, W.: Real-time delay estimation based on delay history. Manufacturing and Service Operations Management **11**(3) (2009) 397–415
16. Schruben, L., Kulkarni, R.: Some consequences of estimating parameters for the m/m/1 queue. Operations Research Letters **1**(2) (1982) 75 – 78
17. Brown, L., Gans, N., Mandelbaum, A., Sakov, A., Shen, H., Zeltyn, S., Zhao, L.: Statistical analysis of a telephone call center. Journal of the American Statistical Association **100**(469) (2005) 36–50
18. Whitt, W.: Stochastic-process limits: an introduction to stochastic-process limits and their application to queues. Springer (2002)
19. Nguyen, V.: The trouble with diversity: Fork-join networks with heterogeneous customer population. The Annals of Applied Probability (1994) 1–25
20. Gans, N., Koole, G., Mandelbaum, A.: Telephone call centers: Tutorial, review, and research prospects. Manufacturing & Service Operations Management **5**(2) (2003) 79–141
21. van der Aalst, W., Nakatumba, J., Rozinat, A., Russell, N.: Business process simulation: How to get it right. BPM Center Report BPM-08-07, BPMcenter. org (2008)
22. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic Petri nets with arbitrary firing delays. In: ICSOC. Vol 8274 of LNCS, pages 389-403, 2013.
23. Woodside, C.M., Stanford, D.A., Pagurek, B.: Optimal prediction of queue lengths and delays in gi/m/m multiserver queues. Operations Research **32**(4) (1984) pp. 809–817
24. Whitt, W.: Predicting queueing delays. Management Science **45**(6) (1999) 870–888
25. Folino, Francesco, Massimo Guarascio, and Luigi Pontieri: Discovering context-aware models for predicting business process performances. In: On the Move to Meaningful Internet Systems: OTM 2012. Springer Berlin Heidelberg, 2012. 287-304.