Discovery and Validation of Queueing Networks in Scheduled Processes

Arik Senderovich[1], Matthias Weidlich[2(✉)], Avigdor Gal[1], Avishai Mandelbaum[1], Sarah Kadish[3], and Craig A. Bunnell[3]

[1] Technion – Israel Institute of Technology, Haifa, Israel
sariks@tx.technion.ac.il, {avigal,avim}@ie.technion.ac.il
[2] Imperial College London, London, UK
m.weidlich@imperial.ac.uk
[3] Dana-Farber Cancer Institute, Boston, MA, USA
{sarah_kadish,craig_bunnell}@dfci.harvard.edu

Abstract. Service processes, for example in transportation, telecommunications or the health sector, are the backbone of today's economies. Conceptual models of such service processes enable operational analysis that supports, e.g., resource provisioning or delay prediction. Automatic mining of such operational models becomes feasible in the presence of event-data traces. In this work, we target the mining of models that assume a resource-driven perspective and focus on queueing effects. We propose a solution for the discovery and validation problem of scheduled service processes - processes with a predefined schedule for the execution of activities. Our prime example for such processes are complex outpatient treatments that follow prior appointments. Given a process schedule and data recorded during process execution, we show how to discover Fork/Join networks, a specific class of queueing networks, and how to assess their operational validity. We evaluate our approach with a real-world dataset comprising clinical pathways of outpatient clinics, recorded by a real-time location system (RTLS). We demonstrate the value of the approach by identifying and explaining operational bottlenecks.

1 Introduction

Service systems play a central role in today's economies, e.g., in transportation, finance, and the health sector. Service provisioning is often realized by a *service process* [1,2]. It can be broadly captured by a set of activities that are executed by a service provider and designated to both attain a set of organizational goals and add value to customers.

Independently of the domain, service processes can be classified by the amount of interactions between service providers and customers and the level of demand predictability and capacity flexibility. A service can be *multi-stage*, in the sense that service provisioning involves a series of interactions of a customer with a provider, or specific resources at a provider's end. Further, a process can be *scheduled*, meaning that the number of customers to arrive is known in advance, up

J. Zdravkovic et al. (Eds.): CAiSE 2015, LNCS 9097, pp. 417–433, 2015.
DOI: 10.1007/978-3-319-19069-3_26

to last moment cancellations and no-shows. Then, customers follow a pre-defined series of activities, with every activity having a planned starting time for its execution, a duration, and a set of involved resources. Multi-stage scheduled processes are encountered, for instance, in outpatient clinics, where various types of treatments are provided as a service to patients [3]. Here, a schedule determines when a patient undergoes a specific examination or treatment. Another example of multi-stage scheduled processes is public transportation, where schedules determine which vehicle serves a certain route at a specific time [4].

In this work, we focus on the following operational question for multi-stage scheduled service processes: *how to assess the conformance of the schedule of a service process to its actual execution?* To address this question, we develop an approach that is based on discovery and validation of resource-centered models. First, we discover a deterministic model that represents a planned execution of a service process (schedule). Second, we check the conformance of a schedule against a log of recorded process executions.

Our choice of formalism to capture a resource-centered view of service processes is driven by two challenges that arise from multi-stage scheduled service processes, namely dependencies and synchronization. Specifically, in multi-stage processes, customers go through a complex network of resources prior to service completion. Hence, resource demand does not arrive at random, and dependencies between arrivals at different resources must be taken into account. Second, in scheduled processes, customers are delayed not only due to scarce capacity of providers (*resource queues*), but also in *synchronization queues* where customers, for whom activities are executed concurrently, experience delays before they can proceed towards subsequent service phases. The formalism that we select captures the resource perspective in service processes is *queueing networks* and more specifically, *Fork/Join networks* [5]. The Fork/Join networks allow for performance analysis and optimization of parallel queueing systems [35]. Adopting this formalism, the contribution of this paper is summarized as follows:

(1) We propose a technique to discover a deterministic Fork/Join network from a *schedule* of a service process.
(2) We present a conformance checking technique that assesses the validity of a deterministic Fork/Join network based on recorded process executions.

We demonstrate the value of the proposed discovery and validation methods by applying them to RTLS-based data from a real-world use-case of scheduling in a large outpatient oncology clinic. Our experiments demonstrate the usefulness of the proposed methods in detecting operational bottlenecks in the schedule, specifically longer-than-planned synchronization delays, and diagnosing the root-cause to those problems.

The remainder of the paper is structured as follows. Section 2 presents a detailed use-case of a process in an outpatient clinic. The models for schedules and log data of service processes is outlined in Section 3. Fork/Join networks and their discovery from a schedule are discussed in Section 4. The technique to check conformance of schedule-driven models against recorded process executions is presented in Section 5. An empirical evaluation of our approach is given in

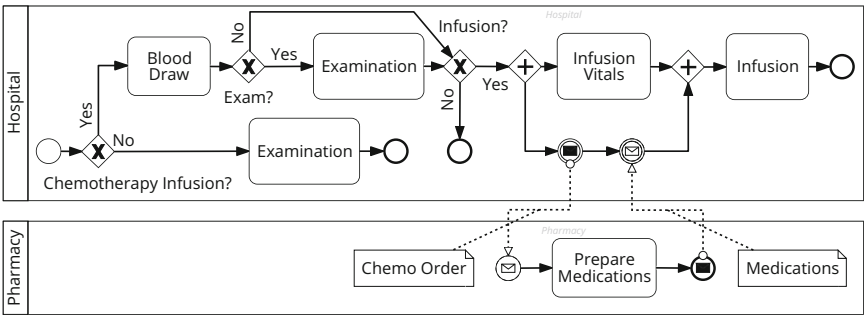

Fig. 1. The control-flow perspective of patient flow in the day hospital

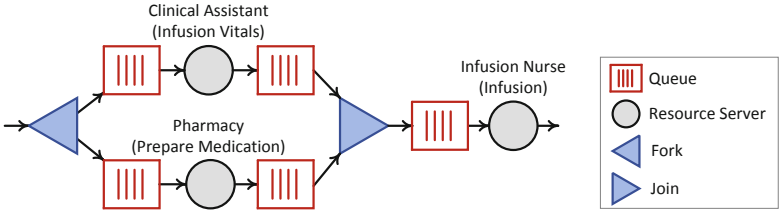

Fig. 2. A Fork/Join network that depicts the scheduled process from the resource perspective

Section 6. Section 7 discusses related work, followed by concluding remarks and future work (Section 8).

2 A Service Process in an Outpatient Clinic

We illustrate the challenges that arise from operational analysis of multi-stage scheduled service processes through a process in the Dana-Farber Cancer Institute (DFCI), a large outpatient cancer center in the US. In this hospital, approximately 900 patients per day are served by 300 health care *providers*, e.g. physicians, nurse practitioners, and registered nurses, supported by approximately 70 administrative staff. The hospital is equipped with a Real-Time Location System (RTLS). We use the movements of patients, personnel, and equipment recorded by this system to evaluate our approach.

We focus on the service process for a particular class of patients, the on-treatment patients (OTP). This process applies to 35% of the patients, yet it generates a large fraction of the workload due to the long service times. Hence, operational analysis to balance quality-of-service and efficiency is particularly important for this process. Figure 1 depicts the control-flow perspective of the process as a BPMN diagram: Arriving patients may directly receive examination by a physician, or shall undergo a chemotherapy infusion. For these patients, a blood draw is the initial appointment. Then, they either move to the infusion stage directly, or first see a provider for examination.

For a specific part of the aforementioned chemotherapy infusion process, Fig. 2 illustrates a queueing network that captures the resource perspective of the process. This model is a Fork/Join network, discussed in more detail in Section 4. It represents the associated resources: clinical assistants, a pharmacy, and infusion nurses, as well as dependencies between them that follow from the patient flow. Patients first fork and enter two *resource queues* in parallel: one is the queue where they actually sit and wait for a clinical assistant to take their vital signs; the other queue is virtual, where they wait for their chemotherapeutic drugs to be prepared by the central hospital pharmacy. The process can only continue once both of these parallel activities are completed, which explains the existence of *synchronization queues* in front of the join of the flows. After the join, patients are enqueued to wait for a nurse and chair to receive infusion.

The provisioning of infusions in DFCI is scheduled. Therefore, each patient has a schedule that assigns a planned time for execution by the respective activities. As such, a schedule allows not only for discovery of the structure of a Fork/Join network, such as the one in Fig. 2, but also its annotation in terms of arrival rates, service times, and server capacities. Such a *schedule-driven model* reflects the performance of the *planned* execution of the process, which raises the question whether the model accurately reflects the actual process execution. Any deviation from the plan, e.g., because of aborted activities or differences in resource scheduling policies, negatively impacts the validity of the schedule-driven model. In the presence of recorded process executions, however, conformance checking as presented in this work can be used to assess the behavioral, conceptual, and operational validity of the schedule-driven model.

3 Schedules and Event Logs of Service Processes

In this work, we provide a multi-level analysis approach that exploits two types of input data, namely a *schedule* and a *log* of recorded actual executions of activities. Such multi-level representation enables analysis that is richer than the state-of-the-art. Below, we formalize the models for schedules and event logs.

Schedule. A schedule represents the plan of a multi-staged service process for *individual customers*. It comprises *tasks* that are partially ordered. We define a task to be a relation between customers, activities, resources at a time, for a duration, e.g., customer *Smith* is to perform a *blood draw* with nurse *Greenberg* on Monday, 03/02/2016 8:00, for 5 minutes. We denote the universe of tasks by T.

Definition 1 (Schedule)**.** *A* schedule *is a set of planned tasks, $T_P \subseteq T$, having a schema (set of functions) $\sigma_P = \{\xi_p, \alpha_p, \rho_p, \tau_p, \delta_p\}$, where*

- $\xi_p : T \to C$ *assigns a customer to a task.*
- $\alpha_p : T \to A$ *assigns an activity to a task.*
- $\rho_p : T \to R$ *assigns a resource to a task.*
- $\tau_p : T \to \mathbb{N}^+$ *assigns a timestamp representing the planned start time to a task.*

– $\delta_p : T \to \mathbb{N}^+$ *assigns a duration to a task.*

The timestamps assigned by τ_p *induce a partial order of tasks, denoted by* $\prec_P \subseteq T_P \times T_P$.

Log. A log contains the data recorded during the execution of the service process, e.g., by a Real-Time Location System (RTLS) as in our use-case scenario (Section 2). Task events in the log relate to a customer, resource, activity, and the timestamps of execution, thereby representing a unique instantiation of an activity executed by a resource for a customer at a certain time.

Definition 2 (Log). *A* log *is a set of sequences of executed tasks,* $T_A \subseteq T^*$, *having a schema* $\sigma_A = \{\xi_a, \alpha_a, \rho_a, \tau_{start}, \tau_{end}\}$, *where*

– $\xi_a : T \to C$ *assigns a customer to a task.*
– $\alpha_a : T \to A$ *assigns an activity to a task.*
– $\rho_a : T \to R$ *assigns a resource to a task.*
– $\tau_{start} : T \to \mathbb{N}^+$ *assigns a timestamp representing the observed start time to a task.*
– $\tau_{end} : T \to \mathbb{N}^+$ *assigns a timestamp representing the observed end time to a task.*

4 Discovering Queueing Networks for Scheduled Processes

The approach taken in this paper is outlined in Figure 3. We first discover a deterministic F/J network from a schedule (see Definition 1). Then, data-driven techniques create an enriched F/J network that is used for validation. This section focuses on the discovery stage, by first defining Fork/Join networks (Section 4.1), before elaborating on the actual discovery (Section 4.2). Enrichment and validation are detailed in Section 5.

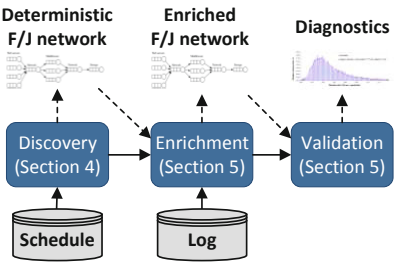

Fig. 3. An outline of our approach

4.1 F/J Queueing Networks: Structure and Dynamics

Queueing networks are directed graphs, with each node corresponding to either a server (a resource) or a queue, see Fig. 2. We consider single-class (customers are of one type), open (customers arrive from outside the system and depart eventually), Fork/Join (F/J) queueing networks [5]. F/J networks extend 'classical' queueing networks with support for splitting and joining of customer instances, which makes them particularly suited to model concurrent actions in service processes.

To define F/J networks, we first need to specify server dynamics. To capture these dynamics, we adopt a version of Kendall's notation [8], so that every server

is characterized by, $\mathcal{A}/\mathcal{B}/\mathcal{C}+\mathcal{P}$ where $\mathcal{A}$ is the arrival rate for queues with *external* arrivals, given as the joint distribution of inter-arrival times; $\mathcal{B}$ is the service time of processing a single customer, given as the distribution of this time; and $\mathcal{C}$ is the capacity, given as the number of resources. Note that arrivals, service times, and resource capacity may be defined in a time-varying manner. Further, component $\mathcal{P}$ is the service policy that sets both the order of entry-to-service, among the enqueued customers, and selects the resource, among available resources, to serve a customer. For example, the most well-known service policy for queues is the First-Come First-Served Policy (FCFS), which is often combined with a server selection strategy that choses the first server that becomes available.

F/J networks support two types of queues: *resource queues* are formed because of limited resource capacity; *synchronization queues* result from simultaneous processing by several resources. Servers can be of three types: regular (services with capacity), fork, and join. A routing matrix defines the flow between servers and queues. With $\mathcal{K}$ being the universe of possible dynamics models for a server, we define F/J network as follows.

Definition 3 (F/J Network)**.** *An* F/J network *is a quadruple* $\langle S, Q, W, b\rangle$*, where*
- $S = S_R \cup S_F \cup S_J$ *is a set of servers, with* S_R *being a set of resources and* S_F, S_J *being sets of forks and joins, respectively;*
- $Q = Q_R \cup Q_S$ *is a set of queues, with* Q_R *being a set of resource queues and* Q_S *being a set of synchronization queues;*
- $W : (Q \times S) \cup (S \times Q) \to [0, 1]$ *is a routing matrix that assigns weights (or probabilities) to edges between servers and queues;*
- $b : S \to \mathcal{K}$ *assigns a dynamics model to servers.*

We consider forks and joins to be zero-delay, single-server nodes. Further, the entries of the routing matrix for forks and joins are binary. For a fork $s \in S_F$, a customer always continues in *all* downstream queues $q \in Q$ for which $W(s, q) = 1$. For a join $s \in S_J$, a customer needs to be selected from *all* upstream queues $q \in Q$ with $W(q, s) = 1$ before continuation. An F/J network for which the complete routing matrix is binary is *deterministic*, otherwise the network is *probabilistic*.

As an example, consider the F/J network in Fig. 2. It contains three resources, a fork, a join, three resource queues (preceding resources), and two synchronization queues (succeeding resources). Note that the structure given in Fig. 2 can only be a deterministic F/J network as all queues and servers, which are not forks, have a single outgoing edge.

4.2 Discovering a Scheduled Queueing Network

Under the assumption that the process follows the schedule in terms of the execution of activities for customers by resources, i.e., there may only be temporal deviations, the network structure is directly derived from the schedule. The timestamps of the schedule tasks in T_P induce a partial order $\prec_P$, which gives raise to a dependency graph between activities and, thus, resource nodes. The structure of the F/J network is then created by inserting the required forks and

joins, at resource nodes with more than one predecessor or successor, respectively. Further, resource queues are inserted for all incoming edges of resource nodes, which are no forks or joins. Finally, synchronization queues are added for all incoming edges of joins.

To extract the dynamics models, for each resource node $s \in S_R$, we shall assume that the number of resources as function of time, $K_s(t)$, can easily be extracted from the schedule log (see also [9] for a respective method). To extract the arrivals ($\mathcal{A}$), service times ($\mathcal{B}$), and service policy ($\mathcal{P}$) from a schedule of tasks T_P with schema $\sigma_P = \{\xi_p, \alpha_p, \rho_p, \tau_p, \delta_p\}$, partially ordered by $\prec_P$, we proceed as follows.

Arrival Times. For each of the most upstream resource nodes $s \in S_R$, arrival times

$$A(s, T_P, \sigma_P, \prec_P) = \{\tau_p(t) \mid t \in T_P \ \wedge \ \rho_p(t) = s \ \wedge \ \forall\, t' \in T_P : \xi_p(t) = \xi_p(t') \Rightarrow t' \not\prec_P t\}, \tag{1}$$

are defined by the first event in the schedule for the planned process execution for each customer. Arrivals into downstream resource nodes are assumed to correspond to end-of-processing times of previous nodes.

Service Times. For a resource node $s \in S_R$, services times

$$B(s, T_P, \sigma_P, \prec_P) = \{\delta_p(t) \mid t \in T_P \ \wedge \ \rho_p(t) = s\}, \tag{2}$$

are defined by the planned duration of the respective activity.

Service Policy. A service policy is a, potentially time-dependent, function that selects a task to be served from a set of waiting tasks. A typical schedule assumes that tasks are served according to an *earliest-due-date* (EDD) policy with the additional constraint that tasks arriving ahead of their due time do not get served. Formally, given a set of waiting tasks, $\{t_1, \ldots, t_n\}$ at time t, the EDD policy is given by:

$$P(\{t_1, \ldots, t_n\}, t) = \text{argmin}_{t' \in \{t_1, \ldots, t_n\}, \tau_p(t') < t}\{\tau_p(t')\}. \tag{3}$$

5 Validating Schedule-Driven Networks with Logs

A F/J network discovered from a schedule enables operational analysis of the planned process behavior under the assumption that there are no deviations from the plan. In most cases, however, such deviations are likely to be observed, which calls for validation of the schedule-driven model. In the presence of a log of recorded process executions, such validation can be achieved by conformance checking.

Below, we first review dimensions of validity for operational models, i.e., behavioral, operational, and conceptual validity (Section 5.1). Given the rich body of literature on methods to ensure behavioral validity, we focus on the other dimensions and present a methodology to ensure operational validity (Section 5.2) and conceptual validity (Section 5.3). For both dimensions, we also instantiate the methodology and outline methods to validate specific model aspects, i.e., processing delays and service policies. Finally, we elaborate on the link between conceptual and operational validity (Section 5.4).

5.1 Dimensions of Validity

Validity relates to *behavioral*, *operational*, and *conceptual* model aspects [10].

Behavioral Validity. Given a process model, its behavioral relevance with respect to the real process is of crucial importance. Four complementary notions of behavioral validity are given in the literature, namely *fitness*, *simplicity*, *precision* and *generalization* [11]. Various techniques to assess behavioral validity have been proposed, among some *trace replay* [12], also known as trace-based simulation [7], *trace alignment* [13], the comparison of behavioral relations [14], and the injection of negative events [15].

Deterministic F/J networks are equivalent to decision-free Petri-Nets [16]. Therefore, the aforementioned techniques to assess behavioral validity can directly be lifted to the resource perspective of a process that is given as a deterministic F/J network.

Operational Validity. Operational validity concerns model performance measures and the accuracy of conclusions drawn from it. Operational models and recorded executions of a process may be consistent in terms of ordering, i.e., the model is behaviorally valid. However, the same model may show low operational validity and be inaccurate in the operational sense. An example would be too coarse-grained abstractions, e.g., a schedule consisting of a single activity per customer and logs that record only the execution of this activity. The discovered model, while behaviorally valid, may be useless for operational analysis. Specifically, execution times for the single activity that would result from the model may not match their corresponding times from the log, because of their large variability that stems from the aggregation of customers that are served differently.

Conceptual Validity. Conceptual validity is defined as the checking of the assumptions and theories that underlie a model [10]. For the context of F/J networks, conceptual validity relates to its structure and the dynamics models of server nodes. Operational models are typically built with assumptions related to the distribution of execution times, customer arrival rates, and routing probabilities. Moreover, some operational models rely on approximations and therefore, should be tested for their applicability against data. To determine conceptual validity, various techniques can be used, for example, statistical tests can be applied to verify model assumptions and quantify the deviation between assumed values (e.g., first-moments) and actual measurements (data) [10].

5.2 Operational Validity: Detecting and Quantifying Performance Deviations

Methodology for Operational Validity. To assess operational validity of a model, model-based performance measures are compared to their counterparts in the recorded log data. The specific measures to consider may vary among scenarios. However, the comparison between the model and the recorded data

typically exploits a discrepancy metric between the two, based on a specific performance measure (or output).

Formally, let $D : O \times O \to X$ be the deviation between an output of a model and the corresponding actual measurements (or the output of another model), where O is the output domain (e.g., processing delays) and X is the domain of the measure for deviation (e.g. the distance between processing delays is also a delay).

Once the performance measures and distance function are set, one can apply a replay technique, see [12], to collect sample values of D for every planned task $t \in T_P$ that also exists in the log, i.e., $t \in T_A$. The replay will result in sample of the deviations between two outputs, $\{D(t) \mid t \in (T_P \cap T_A)\}$. This sample can, in turn, be analyzed via statistical techniques for measuring goodness-of-fit between two models.

Validation of Processing Delays. We now turn to an instantiation of the methodology to assess operational validity for a specific model aspect. We take up the aforementioned case of Dana-Farber, where quality improvement teams focus on *delays* of individual patients per activity with respect to the maximum between the scheduled time and the time of arrival at the respective resource. To investigate this aspect, a distribution is constructed from the individual delays. Then, the difference between the measure from the log and its equivalent from the deterministic F/J network is quantified.

Using a continuous time model, the specific deviation function for processing delays is defined as $D_D : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. For convenience, we assume $T_A \subseteq T_P$, i.e., no unplanned tasks arrived to the system. While this assumption is not necessary for the demonstrated approach, it simplifies the definitions in the remainder.

Delays in processing may be caused by resource capacity (customers wait in a resource queue) or synchronization (customers wait in a synchronization queue). Hence, for a task $t \in T_A$, there is a resource delay $\widehat{W_R(t)}$ and a synchronization delay $\widehat{W_S(t)}$, and the total delay is their sum, $\widehat{W(t)} = \widehat{W_R(t)} + \widehat{W_S(t)}$. The resource delay is the difference between the current time and the maximum between synchronization time between previous tasks and the scheduled time for the task. The synchronization delay is the difference of the earliest and latest entry in one of the synchronization queues.

From the schedule, for resource $s \in S_R$, we extract the planned delay $W(t, s)$ as the timestamp difference between the task and its direct successors. Then, we define $D_D(t) = D(\widehat{W(t)}, W(t))$ as one of the well-established metrics for deviations between two outputs, e.g. the squared deviation. This way, deviations in processing delays between the schedule and the observed process execution are detected and quantified.

5.3 Conceptual Validity: Checking Model Assumptions

Methodology for Conceptual Validity. We approach the conceptual validity of a deterministic F/J network discovered from the schedule by means of *enhancement* and *comparison*. That is, the deterministic F/J network is enhanced based

on the log data, which yields a stochastic F/J network. Conceptually, this step is similar to enhancement operations known for process models that focus on the control-flow perspective, see [17]. As a by-product of applying the enhancement algorithm, however, our approach directly compares the two models, specifically, the server dynamics of the deterministic, schedule-driven model to the dynamics of the stochastic, data-driven model.

An *enhancement function* creates a stochastic F/J network from a deterministic F/J network and a log. In practice, enhancement boils down to fitting all model elements of the stochastic F/J network with the log data, namely identifying the model structure and, for every server, discovering its dynamics. Generally speaking, enhancement combines several process mining and statistical techniques:

- To extract the model structure, process discovery algorithms that exploit direct successorship of activities and detect concurrency can be used, e.g., the family of α-algorithms [18, Ch.5].
- The routing matrix can be inferred by its empirical equivalent, i.e. counts over sums of historical transitions between nodes.
- Service policies for routing customers can be discovered using the policy-mining techniques presented in [19].
- The distribution of inter-arrival and service times can be fitted via techniques that were developed and applied in [20,21].

Differences in the structure and server dynamics of the schedule-driven model and the data-driven model are assessed by statistical comparison techniques, e.g., hypothesis testing [22]. This allows for quantifying deviations and concluding on their significance.

Validation of Service Policies. Next, we instantiate the methodology and demonstrate a statistical comparison method for service policies that determine the routing of customers to resources. We focus on a single resource node $s \in S_R$ and assume that it has been previously diagnosed (via the techniques to assess operational validity) to cause delays downstream. Hence, we aim at comparing that server's service policy and verify whether the schedule-driven policy indeed holds.

Formally, let P be the assumed policy that has to be checked against all historical decisions on the next customer to enter service, which are represented in the log. Policy P supposedly follows Equation 3, i.e., for a set of tasks $\{t_1, \ldots, t_n\}$ waiting in the respective resource queue at time t, the task that has the earliest scheduled timestamp is selected. To assess to which extent this policy holds true, we define a respective indicator $\mathbb{1}_{P(i)}$, which is equal to one if indeed the i-th past decision corresponds to Equation 3. Then, we define a statistic that quantifies the level of compliance to policy P:

$$\chi_P = \frac{1}{|n|} \sum_{i=1}^{n} \mathbb{1}_{P(i)}, \tag{4}$$

This is an estimate of the probability that P holds, i.e., $\mathbb{E}[\mathbb{1}_P] = \mathbb{P}(P)$ with P being the compliance-to-policy-P event. We use this estimate to test several

plausible policies, e.g., First-Come First-Serve, and decide on the best-fitting policy.

5.4 Continuous Validity

We tighten the relations between conceptual and operational validity by adopting the paradigm of *continuous validity*. This paradigm means that two models that are equivalent in the conceptual domain, will also be equivalent in the operational domain. The notion of conceptual equivalence between two models can be derived from model comparison as described above, while operational equivalence can be defined with respect to our measure for deviations, D (see Section 5.2). The result of continuous validation can be interpreted in two ways. First, assuming that the schedule is the normative process, one should fix the causes for deviation in process executions. Alternatively, if the actual execution is the reference point, the schedule is to be repaired accordingly.

6 Evaluation

This section presents an empirical evaluation of our approach based on the case study of the Dana-Farber Cancer Institute, see Section 2. Specifically, we use the appointment schedule, RTLS data, and pharmacy data to discover a schedule-based queueing network. Then, we demonstrate an operational validation of the model that searches for deviations from the scheduled process in the temporal sense. Last, we check the conceptual validity to locate the root-cause of these deviations.

Below, Section 6.1 describes the datasets and experimental setup. Section 6.2 discuss the obtained discovery and validation results.

6.1 Datasets and Experimental Setup

Our experiments combine three data sources from the Dana-Farber Cancer Institute: an *appointment schedule*, an *RTLS log* recording movements of badges (patients and service providers), and a *pharmacy log* that records checkpoints in the medication-production process. The resolution of the RTLS can be as accurate as 3 seconds, depending on the amount of movement of a badge. From the pharmacy log, we only extracted the start and end events for the production process, since we consider the pharmacy as a separate server. The experiments involve three weekdays, April 14-16, 2014, which are days of 'regular' operation (approximately 600 OTP patients) as was verified with local contacts.

Our experiments involved the following steps. First, we discovered a deterministic F/J network from the schedule. Second, we performed an operational validation of the model against log data, with a focus on deviations in processing delay. Third, we assessed conceptual validity. We enhanced the model to obtain a stochastic, data-driven F/J network and then focus on one of the server

nodes, for which operational deviations had been detected in the second step, and analyzed its dynamics.

We implemented our experiments in two software frameworks, SEEStat and SEEGraph. Both tools have been developed in the Service Enterprise Engineering lab,[1] and enable, respectively, statistical and graphical analysis of large operational datasets. In particular, they enable the creation of new procedures for server dynamics (SEEStat), and for the discovery of structure and routing in queueing networks (SEEGraph).

6.2 Results

Model Discovery. As a first step, we discovered a deterministic schedule-driven F/J network. An excerpt of the result is presented in Fig. 4. Note that the excerpt shows only the activities that are conducted by staff of the outpatient clinic, i.e., the preparation of medications by the pharmacy is not shown[2]. Note that the SEEGraph notation for the queueing network distinguishes two types of customer delays, i.e., time spent before the scheduled time (*Planned Delay*) and the processing delay after the time the customer was scheduled (*Waiting Lab/Vitals/Infusion*).

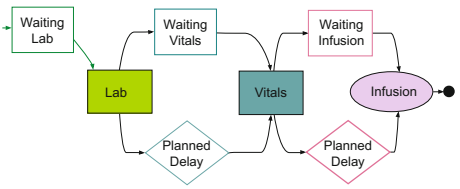

Fig. 4. Schedule-driven process (appointments) extracted from the RTLS data

Operational Validity: Delay Deviations. Next, we enacted the operational validation and tracked down deviating performance measures. Here, we provide the results for the example that is outlined in Fig. 2: a patient that is scheduled to enter infusion waits for two concurrent activities, namely pre-infusion vital signs (vitals) and medication production. The scheduled time between the end of vitals and the beginning of infusion is actually zero and most of the delay is planned for the beginning of vitals.

Figure 5 presents the actual distribution of time between vitals and the beginning of infusion, based on the RTLS data. We observe that, indeed, a large portion of patients go into infusion within a minute from vitals. However, the distribution presents a long tail of patients, who wait for the next step (average delay of 23 minutes). For most patients, this is due to synchronization delays since they wait for their medications. In many occasions, one can observe in the RTLS data that patients wait, while infusion nurses are available for service. This again points toward synchronization delays between the vitals activity and the pharmacy. According to schedule, the central pharmacy is planned to deliver the medication in synchronization with vitals (within 30 minutes). The operational insight of long synchronization delays, however, hints at a conceptual issue regarding the just-in-time arrival of the medication.

[1] http://ie.technion.ac.il/labs/serveng/

[2] An animation can be found at http://youtube.com/watch?v=ovXu3DB9RuQ

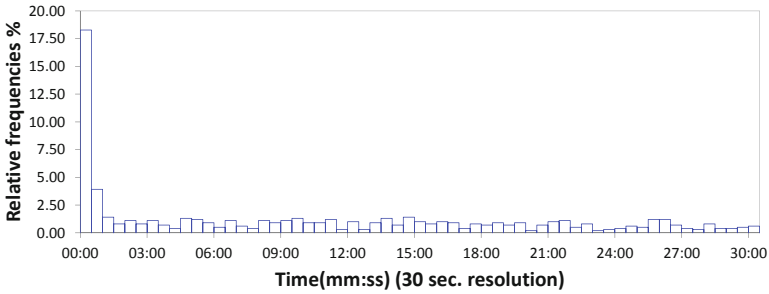

Fig. 5. Waiting time for Infusion (after vitals); Sample size = 996, Mean = 25min, Stdev = 29min

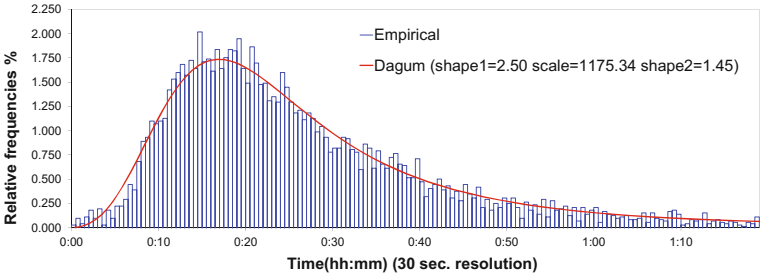

Fig. 6. Medication production time; Sample size = 7187, Mean = 30min, Stdev = 24min

Conceptual Validity: Production Times and Policy. To check for a conceptual issue related to the 'Pharmacy' resource, we investigated its dynamics. We assume that the fork in Fig. 2 is zero-delay and that pharmacy is notified once the patient is ready for infusion. Therefore, we assume that the arrival times do not deviate, and diagnose the two remaining dynamics: production time and service policy. Figure 6 shows the distribution of production times (for April 2014), and the fitted 'Dagum' distribution. Here, we observe that the stochastic model shares a first moment with the planned duration: both are 30 minutes on average. Therefore, in alignment to our continuous-validity paradigm, the root-cause for operational deviations is not the length of drug production.

We now turn to the second dynamic component, the service policy for the drug production. Here, we focus on the time until the first drug has been prepared. Although patients often require more than one drug, the first medication is the one that is needed for the process to flow. Using the method proposed in Section 5.3, we estimated the expected indicators for three policies: (1) Earliest-Due-Date (EDD) First, which corresponds to the plan, (2) First-Come First-Served (FCFS), which produces according to the order of prescription arrivals and (3) Shortest Processing Time (SPT) first, which implies that priority will be given to patients with shorter infusion durations.

Figure 7 presents the estimated proportion of compliance to policy, as a function of the number of medication orders in queue. To see an effect of selection

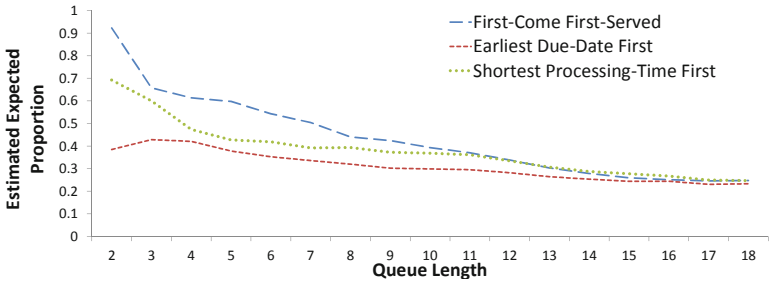

Fig. 7. Policy comparison for the pharmacy resource

based on a policy, the comparison starts with a queue of size two. We observe that as the queue grows, the decision on the next task to enter service becomes more random. However, for short queues, the selection policy tends towards FCFS, instead of EDD as assumed in the schedule. The deviation between the two policies, planned and actual, can be seen as a cause of the synchronization delays observed in Fig. 5.

7 Related Work

Recently, there has been an increased interest in scheduled service processes, especially in the health sector. Outpatient clinics that operate as a scheduled multi-stage process, are of particular interest, due to their pervasiveness and growth over the past years [3]. Performance questions for scheduled multi-stage processes relate to bottleneck identification, dynamic resource allocation, and optimal control (decision making).

Traditionally, such performance analysis is based on techniques from Operations Research disciplines, such as Scheduling [23] and Queueing Theory [24]. These methods use hand-made (highly abstract) models of the reality, and apply the relevant (model-specific) analysis. Validation of the results is typically performed by simulating the 'reality' (again a hand-made model), and comparing the outputs of the modeled reality and the simulated reality, neglecting the benefits of data-driven validation, c.f. [25].

The rapidly evolving field of *process mining*, in turn, is driven by the available data [18]. Models are discovered from and validated against event data that stems from recorded process executions, see [26]. Mined models are used as the basis for prediction [27,28], simulation [17], and resource-behavior analysis [29,30]. However, much work in this field focuses on the *control-flow perspective*, i.e. the order of activities and their corresponding resources, times and decisions [18, Ch.8], so that the created models cannot benefit from the analysis techniques developed in Operations Research. In earlier work, therefore, we argued for an explicit representation of the *queueing perspective* and demonstrated its value for several real-world processes [9,19]. However, the existing techniques all considered the simplistic setting of a single-station system, whereas, this paper addressed the more complex scenario of service processes that are scheduled and have a multi-stage structure that involves resource synchronization.

Our approach of discovering a model from a schedule is similar to the transformation of schedules to Petri-Nets presented in [31]. However, our target formalism is Fork/Join networks to leverage existing analysis techniques for queueing networks. Also, we employ log data to answer structural and operational questions regarding the schedule.

One of the main questions in scheduled processes is that of conformance of the actual process execution to the plan. Techniques for model validation in process mining primarily focus on behavioral validity, see [12–15]. However, a few works also addressed time and resource validity of discovered models [32,33], where the replay method, as in [34], is used to quantify deviations in performance measures. However, in these approaches, conceptual validity (model assumptions) is confounded with operational validity (resulting performance measures). This paper argues for a clear separation between behavioral, operational and conceptual validity, and introduces a methodology for assessing the operational and conceptual validity of Fork/Join networks.

8 Conclusion

In this work, we provided a framework for the operational analysis of scheduled multi-stage service processes, as they are observed in such domains as healthcare and transportation. To assess the conformance of the schedule of a process and its actual execution, we presented an approach based on discovery and validation of queueing networks. First, we showed how a deterministic Fork/Join network is discovered from a schedule. Second, we presented a method that exploits log data to assess the operational and conceptual validity of the discovered model. We evaluated the approach with real-world data from an outpatient clinic and showed how our approach leads to the identification of operational bottlenecks and supports the analysis of the root-causes of these bottlenecks.

In future work, we would like to test the value of the stochastic Fork/Join network in the context of queue mining, e.g. for delay prediction in a network. Also, we hope to extend our validation techniques to incorporate features of stochastic analysis, when comparing two models, e.g., by developing similarity measures for Fork/Join networks.

Acknowledgments. This work was supported by the EU INSIGHT project (FP7-ICT 318225). We are grateful to the SEELab members, Dr. Valery Trofimov, Igor Gavako and especially Ella Nadjharov, for their help with the statistical analysis. We also thank Kristen Camuso, from Dana-Faber Cancer Institute for the insightful data discussions.

References

1. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013)
2. Daskin, M.S.: Service Science. Wiley. com (2011)

3. Froehle, C.M., Magazine, M.J.: Improving scheduling and flow in complex outpatient clinics. In: Handbook of Healthcare Operations Management, pp. 229–250. Springer (2013)
4. Gal, A., Mandelbaum, A., Schnitzler, F., Senderovich, A., Weidlich, M.: Traveling Time Prediction in Scheduled Transportation with Journey Segments. Tech. Rep., Technion (2014)
5. Ammar, M.H., Gershwin, S.B.: Equivalence relations in queueing models of fork/join networks with blocking. Performance Evaluation **10**(3), 233–245 (1989)
6. Mandelbaum, A.: Service engineering (science, management): A subjective view. Technical report, Technion-Israel Institute of Technology (2007)
7. 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)
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), 338–354 (1953)
9. 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)
10. Sargent, R.G.: Verification and validation of simulation models. In: WSC, pp. 183–198 (2011)
11. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Quality dimensions in process discovery: The importance of fitness, precision, generalization and simplicity. Int. J. Cooperative Inf. Syst. **23**(1) (2014)
12. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)
13. Bose, R.P.J.C., van der Aalst, W.M.P.: Process diagnostics using trace alignment: Opportunities, issues, and challenges. Inf. Syst. **37**(2), 117–141 (2012)
14. Weidlich, M., Polyvyanyy, A., Desai, N., Mendling, J., Weske, M.: Process compliance analysis based on behavioural profiles. Inf. Syst. **36**(7), 1009–1025 (2011)
15. vanden Broucke, S.K.L.M., Weerdt, J.D., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data Eng. **26**(8), 1877–1889 (2014)
16. Dallery, Y., Liu, Z., Towsley, D.: Equivalence, reversibility, symmetry and concavity properties in fork-join queuing networks with blocking. J. ACM **41**(5), 903–942 (1994)
17. Rozinat, A., Mans, R., Song, M., van der Aalst, W.M.P.: Discovering simulation models. Information Systems **34**(3), 305–327 (2009)
18. van der Aalst, W.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
19. Senderovich, A., Weidlich, M., Gal, A., Mandelbaum, A.: Mining resource scheduling protocols. In: Sadiq, S., Soffer, P., Völzer, H. (eds.) BPM 2014. LNCS, vol. 8659, pp. 200–216. Springer, Heidelberg (2014)
20. 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), 36–50 (2005)
21. Zhang, P., Serban, N.: Discovery, visualization and performance analysis of enterprise workflow. Computational statistics & data analysis **51**(5), 2670–2687 (2007)
22. Bickel, P., Doksum, K.: Mathematical Statistics: Basic Ideas and Selected Topics. Holden-Day series in probability and statistics, vol. 1. Prentice Hall (2001)

23. Pinedo, M.: Planning and scheduling in manufacturing and services. Springer (2005)
24. Buzacott, J.A., Shanthikumar, J.G.: Stochastic Models of Manufacturing Systems. Prentice Hall, Englewood Cliffs, NJ (1993)
25. Mans, R.S., Russell, N.C., van der Aalst, W.M.P., Moleman, A.J., Bakker, P.J.M.: Schedule-aware workflow management systems. In: Jensen, K., Donatelli, S., Koutny, M. (eds.) Transactions on Petri Nets and Other Models of Concurrency IV. LNCS, vol. 6550, pp. 121–143. Springer, Heidelberg (2010)
26. Rogge-Solti, A., van der Aalst, W.M.P., Weske, M.: Discovering stochastic petri nets with arbitrary delay distributions from event logs. In: Lohmann, N., Song, M., Wohed, P. (eds.) BPM 2013 Workshops. LNBIP, vol. 171, pp. 15–27. Springer, Heidelberg (2014)
27. van der Aalst, W.M.P., Schonenberg, M., Song, M.: Time prediction based on process mining. Information Systems **36**(2), 450–475 (2011)
28. Rogge-Solti, A., Weske, M.: Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 389–403. Springer, Heidelberg (2013)
29. Pika, A., Wynn, M.T., Fidge, C.J., ter Hofstede, A.H.M., Leyer, M., van der Aalst, W.M.P.: An extensible framework for analysing resource behaviour using event logs. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 564–579. Springer, Heidelberg (2014)
30. Nakatumba, J., van der Aalst, W.M.P.: Analyzing resource behavior using process mining. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 69–80. Springer, Heidelberg (2010)
31. Van der Aalst, W.: Petri net based scheduling. Operations-Res.-Spektr. **18**(4), 219–229 (1996)
32. Taghiabadi, E.R., Gromov, V., Fahland, D., van der Aalst, W.M.P.: Compliance checking of data-aware and resource-aware compliance requirements. In: Meersman, R., Panetto, H., Dillon, T., Missikoff, M., Liu, L., Pastor, O., Cuzzocrea, A., Sellis, T. (eds.) OTM 2014. LNCS, vol. 8841, pp. 237–257. Springer, Heidelberg (2014)
33. de Leoni, M., van der Aalst, W.M.P., van Dongen, B.F.: Data- and resource-aware conformance checking of business processes. In: Abramowicz, W., Kriksciuniene, D., Sakalauskas, V. (eds.) BIS 2012. LNBIP, vol. 117, pp. 48–59. Springer, Heidelberg (2012)
34. Adriansyah, A., van Dongen, B.F., van der Aalst, W.M.P.: Conformance checking using cost-based fitness analysis. In: EDOC, IEEE Computer Society, pp. 55–64 (2011)
35. Atar, R., Mandelbaum, A., Zviran, A.: Control of fork-join networks in heavy traffic. In: 50th Annual Allerton Conference on Communication, Control, and Computing, pp. 823–830 (2012)