The ROAD from Sensor Data to Process Instances via Interaction Mining

Arik Senderovich[1(✉)], Andreas Rogge-Solti[2], Avigdor Gal[1], Jan Mendling[2], and Avishai Mandelbaum[1]

[1] Technion – Israel Institute of Technology, Haifa, Israel
sariks@tx.technion.ac.il, {avigal,avim}@ie.technion.ac.il
[2] Vienna University of Economics and Business, Vienna, Austria
{andreas.rogge-solti,jan.mendling}@wu.ac.at

Abstract. Process mining is a rapidly developing field that aims at automated modeling of business processes based on data coming from event logs. In recent years, advances in tracking technologies, e.g., Real-Time Locating Systems (RTLS), put forward the ability to log business process events as location sensor data. To apply process mining techniques to such sensor data, one needs to overcome an abstraction gap, because location data recordings do not relate to the process directly. In this work, we solve the problem of mapping sensor data to event logs based on process knowledge. Specifically, we propose *interactions* as an intermediate knowledge layer between the sensor data and the event log. We solve the mapping problem via optimal matching between interactions and process instances. An empirical evaluation of our approach shows its feasibility and provides insights into the relation between ambiguities and deviations from process knowledge, and accuracy of the resulting event log.

Keywords: RTLS data · Business processes · Optimal matching · Knowledge-driven

1 Introduction

Process mining is a rapidly developing field that aims at automated modeling of business processes based on data coming from event logs [1]. Most process mining techniques assume that the event logs are directly related to the underlying process and contain information on activities, resources, and durations. In recent years, advances in tracking technologies, using e.g., Real-Time Locating Systems (RTLS), put forward the ability to track entities that are involved in process executions such as customers, resources. Currently, these technologies are mainly used for monitoring location of entities. For example, in hospitals, nurses use a real-time map to track patients that are next to enter service. To apply process mining techniques to location data, one needs to overcome an abstraction gap, since sensor data recordings do not relate to the process directly.

S. Nurcan et al. (Eds.): CAiSE 2016, LNCS 9694, pp. 257–273, 2016.
DOI: 10.1007/978-3-319-39696-5_16

In this paper, we propose a *knowledge-driven* approach that facilitates process knowledge for accurately transforming raw sensor recordings that contain locations and timestamps of process entities into standardized event logs that comprise process instances. To this end, we define the notion of *interactions* as an intermediate knowledge layer. These interactions are mined from the sensor data as a set of recordings that contain entities that overlap in time and space. Assuming that interactions correspond to an activity instance, we formulate an *optimal matching* problem that maps interactions to activity labels. The problem is thus formulated as an Integer Linear Program with parameters encoded from existing process knowledge. The resulting interaction-to-activity mapping creates process instances, which completes the construction of an event log. We test our technique with controlled experiments on simulated data, inspired by a real-life healthcare process in an outpatient cancer hospital.

The remainder of the paper is structured as follows. Section 2 presents our data models and defines the ROAD problem. We outline the solution through interaction mining in Sect. 3. The optimal matching problem between interactions and activities is detailed in Sect. 4. We empirically evaluate our approach in Sect. 5. In Sect. 6, we discuss related work, followed by concluding remarks and future work in Sect. 7.

2 Data Models and Problem Statement

In this section we introduce our data models, and present the ROAD problem that leads from raw sensor data to business process instances. To motivate our work, we start with a running example of a real-life healthcare process.

Example 1 (DayHospital). Figure 1 presents a treatment process in DayHospital, a large outpatient cancer hospital. DayHospital treats cancer patients on an ambulatory basis. Specifically, approximately 1000 patients arrive every day and typically go through three activities: a blood draw, an examination, and a chemotherapy infusion. The hospital is equipped with nearly 900 Real-Time Locating System (RTLS) receivers that track all business entities involved in the process (e.g., patients, physicians, nurses) as well as some of the medical devices. The emitted data is recorded in a 3-s resolution, and is currently used only for real-time tracking of process entities and equipment. This example motivates the following data model definitions.

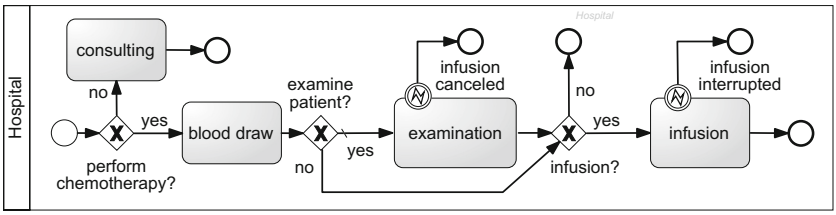


Fig. 1. The main process in an outpatient hospital.

Definition 1 (rO log)**.** *Let B be a set of entity identifiers, $\mathcal{R}$ a set of receiver identifiers, and $\mathbb{TS}$ a set of timestamps. The* raw Observation *(rO) log is a set of triples $D_{raw} = \{(b, r, t)\}$ s.t. $b \in B$, $r \in \mathcal{R}$, and $t \in \mathbb{TS}$.*

Definition 1 formalizes a raw observation log, as captured by an RTLS system. Hereafter, we shall use the "dot" notation to represent elements of a tuple. Therefore, for $d = (b, r, t)$, we use $d.b, d.r$, and $d.t$ to denote the tuple's elements. We assume that an entity can be tracked by a single receiver at a time. As background knowledge we assume the existence of a mapping $\theta : B \rightarrow \mathcal{T}$ that assigns an entity type to an entity identifier. For example, badge identifier 'Bob111' is of type nurse. Entity types may be put into a taxonomy, e.g., an infusion nurse is a type of a nurse. Also, given a tuple $(b, r, t) \in D_{raw}$, we define a spatial function $S : \mathcal{R} \rightarrow \Lambda$, where $\mathcal{R}$ is a set of receiver identifiers as before and Λ is a set of locations that maps receivers into named spatial locations, e.g. Infusion Room 705C.

We next aggregate consecutive reads of the same entity and location as follows. Given an rO log D_{raw}, a *consecutive set of raw observations* is a set $d_{cons} \subseteq D_{raw}$ such that the following three (badge, space and time) continuity conditions hold:

C1: $\forall d = (b, r, t), d' = (b', r', t') \in d_{cons} : b = b' \ \wedge \ S(r) = S(r')$.
C2: $\forall d = (b, r, t), d' = (b', r', t') \in d_{cons}$:
$\exists d'' = (b'', r'', t'') \in D_{raw}(b = b'', S(r) = S(r''), t < t' < t'') \rightarrow d'' \in d_{cons}$.
C3: $\forall d = (b, r, t), d' = (b', r', t') \in d_{cons}$:
$\nexists d'' = (b'', r'', t'') \in D_{raw}(b = b'', S(r) \neq S(r''), t < t'' < t'$.

Condition C1 ensures that every d_{cons} consists of observations that share badge id, $d_{cons}.b$ and location $d_{cons}.l$. C2 states that if badge $d_{cons}.b$ was observed in location $d_{cons}.l$ in the time between two consecutive observations in d_{cons}, that observation also belongs to d_{cons}. Last, C3 ensures that badge $d_{cons}.b$ was not observed in location other than $d_{cons}.l$, in the time between two consecutive observations in d_{cons}. A consecutive set of raw observations d_{cons} is *maximal* if there is no $d'_{cons} \subseteq D_{raw}$ such that $d_{cons} \subset d'_{cons}$. We denote by $D_{cons}(D_{raw})$ (or D_{cons} when D_{raw} is clear from the context) the set of all maximal consecutive sets of raw observations.

Definition 2 (RO log)**.** *Let D_{raw} be an rO log. The* aggRegated Observation *(RO) log over D_{cons} is a set of quadruples $D_{agg} = \{(b, l, s, c)\}$ s.t. $\forall d_{cons} \in D_{cons}$, $\exists d = (b, l, s, c) : b = d_{cons}.b \ \wedge \ l = d_{cons}.l \ \wedge \ s = \min_{d \in d_{cons}}(d.t) \ \wedge \ c = \max_{d \in d_{cons}}(d.t)$.*

The RO log creates a new log from the raw observations log, mapping a receiver identifier to a location and aggregating continuous observations of an entity in a location into intervals with start and end times being the minimum and maximum times over the aggregated raw observations per entity and location, respectively. For the hospital described in Example 1, the RO log is created automatically from a given rO log as a service of the RTLS vendor company.

Table 1. Aggregated hospital tracklog - Sample from Dec. 3rd, 2013.

b	$\theta(b)$	l	s	c
Anna555	Patient	Room 705C	10:00AM	10:30AM
Bob111	Nurse	Room 705C	10:10AM	10:20AM
Anna555	Patient	Room 907	11:50AM	12:17PM
Bob111	Nurse	Room 907	11:40AM	12:15PM
Jenna333	Physician	Room 907	12:00PM	12:20PM

Following Example 1, Table 1 provides a sample of the RO log (the second column associates an entity identifier with an entity type). Patient Anna enters an infusion room 705, chair C at 10:00AM to receive a chemotherapy treatment. She is followed by nurse Bob at 10:10AM, who starts the infusion and leaves the room at 10:20AM. Anna then continues to an examination room with nurse Bob and physician Jenna.

Clearly, the RO log does not contain the necessary information to understand high-level information such as activities, participating resources, and start/end times. For example, we are interested in log entries such as $\langle Anna555, Infusion, \{Bob111\}, InfusionRoom705C, 10{:}10\text{AM}, 10{:}20\text{PM}\rangle$, consisting of patient identifier, activity label, set of resource entities, location, and time interval.

Definition 3 (AD log)**.** *Let B be a set of entity identifiers, A a set of activity labels, Λ a set of locations, and $\mathbb{TS}$ a set of timestamps. The* Activity Data *(AD) log is a set of tuples $L = \{(b, a, E, l, s, c)\}$ that correspond to activity instances s.t. $b \in B$ is the case identifier, $a \in A$ is the activity, $E \subseteq B$ is the set of participating resource entities, $l \in \Lambda$, and $s, c \in \mathbb{TS}$.*

Let $\mathcal{L}$ be the set of all possible AD logs. We define a similarity measure $\Delta : \mathcal{L} \times \mathcal{L} \longrightarrow [0][1]$ that quantifies the extent to which two AD logs differ. Such a similarity measure combines several aspects of the log. For example, when comparing activity labels between traces, we could use a string edit distance measure, while for comparing resource sets we may use Jaccard similarity. The concrete formulation of Δ depends on the requirements of the domain. For example, an organization might be most interested in resource accuracy, while another one might prioritize activity orderings.

Let L be an AD log of a real process, and let α be a mapping such that $\hat{L} = \alpha(D_{agg})$ transforms an RO log into an AD log. Because we do not know the real process, but only observe its RO log D_{agg}, the problem we aim at solving can be states as follows:

Problem 1 (aggRegated Observations to Activity Data (ROAD))**.** *The ROAD problem aims at finding a mapping α of the RO log, D_{agg} to an AD log $\hat{L} = \alpha(D_{agg})$ such that $\Delta(L, \hat{L})$ is minimized.*

3 The ROAD to Solution: Interaction Mining

Our solution to the ROAD problem is based on mining interactions between business entities from the RO log and mapping these interactions to activity instances. Figure 2 depicts our two-step solution to the ROAD problem, which results in a transformation α that maps an RO log to an AD log. While our proposed solution does not provide a formal guarantee of the minimality of Δ, our empirical evaluation verifies that the solution yields accurate results when comparing the AD log ($\alpha(D_{agg})$) with the 'real' event log. We first mine interactions from the RO log (Sect. 3.1). The second step of our approach involves creating an optimal matching between interactions and activities via process knowledge (Sect. 4). This matching results in an AD log.

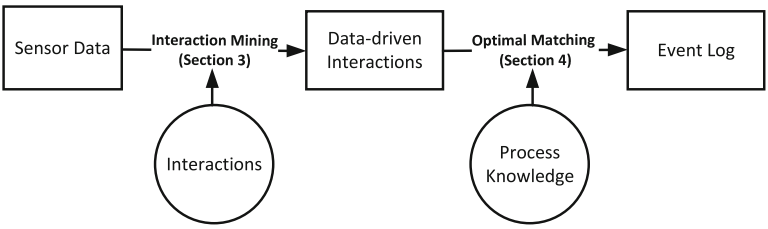


Fig. 2. The ROAD to solution.

To bridge the gap between the RO and the AD logs, we aim at using process knowledge that is readily available in many real-life scenarios, e.g., in the form of process models. However, it is unclear how to directly connect process knowledge and the RO log. For example, observing that a patient is in a certain location does not immediately indicate an ongoing activity. To resolve ambiguities, we create an intermediate knowledge layer, namely *interactions*. These interactions correspond to involvements of business entities, such as patients, nurses, and equipment in activities. To illustrate the notion of interactions, consider Fig. 3 that corresponds to the DayHospital data example from Table 1. Figure 3 depicts the hierarchy of data abstraction levels. Specifically, we assume that activity instances result in interactions, which in turn can be observed in the RO log. In our data example, nurse Bob and patient Anna share a location over time, indicating an interaction that belongs to a certain activity instance, e.g., a chemotherapy infusion.

3.1 Interaction Mining

In this section, we formally define the notion of interaction and propose a methodology for mining interactions from the RO log. The terminology we use follows the terminology of complex event processing (CEP) [2], where complex events are detected from streams of events while our approach operates on historical data logs.

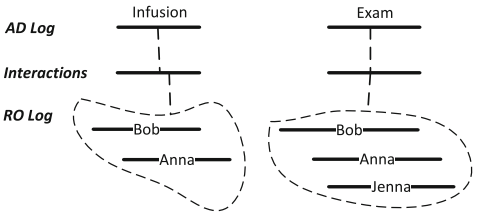


Fig. 3. Hierarchy of Instances: Activities, Interactions, and Raw Data.

Definition 4 (Interaction)**.** *An interaction is a tuple $i = (E, l, s, c)$, where E is a set of interacting entities (badges), l is a location, and s and c stand for the interaction start and end times, respectively.*

Mining a set of interactions $\{(E, l, s, c)\}$ from an RO log requires four basic functions over the RO log, D_{agg}, namely *selection*, *grouping*, *filtering*, and *construction*. As a guiding example we consider the mining of *co-location interactions*, which are interactions that involve two or more entities in the same location over an overlapping timespan. In DayHospital, this a highly relevant interaction, since the execution of medical activities requires the presence of a patient and at least one of the resources.

We first define a selection function over the RO log, which enables us to consider only relevant tuples for interactions. For DayHospital, we are not interested in conference rooms, where doctors spend their time resting, as we are interested in clinical activities.

Definition 5 (Selection)**.** *Let $\psi(\Sigma)$ be a logical predicate over the RO log D_{agg} with a set of external parameters, Σ. A selection function $\sigma(D_{agg}, \psi(\Sigma))$ returns a subset of D_{agg}, s.t. $\{d \in D_{agg} \mid \psi(\Sigma) = True\}$.*

For example, selection over the DayHospital log with location parameter $\lambda = Room705C$ would return all RO tuples in that location. We denote by D_{sel} the outcome of a selection over D_{agg}, which contains individual tuples of the RO log that satisfy the logical rule. However, interactions typically comprise several tuples (e.g. two doctors entering a room correspond to two tuples in the RO log). To this end, we group tuples of D_{sel} into sets by an operation that we refer to as grouping.

Definition 6 (Grouping)**.** *Let $m, M \in \mathbb{N}$ be two natural numbers. A grouping function over D_{sel}, γ, returns the set $\{D \subseteq D_{sel} \mid m \leq |D| \leq M\}$.*

In other words, grouping returns the set of sets of D_{sel} having a minimal size of m, and a maximal size of M. Setting a lower and upper bound on the size of the sets reduces the number of tuple sets, since in the worst case of $m = 1, M = \infty$ one needs to consider all elements of $2^{D_{sel}}$ as interaction candidates. Returning to the co-location example, we are interested in interactions with two entities or

more, thus we set $m = 2$, and $M = \infty$. Let D_{group} denote the grouped set of D_{sel}.

Having gathered all relevant sets of tuples into D_{group} we are interested in filtering out subsets of D_{group} that satisfy an interaction rule (e.g. co-location). For example, if a set in D_{group} contains only tuples that do not share the location, we shall not consider this set as an interaction candidate. To this end, we first define the interaction condition $\eta(D)$ over $D \in D_{group}$, which evaluates to true if D satisfies the condition for an interaction. The filtering function that returns only subsets of D_{group} that satisfy ϕ is defined as follows.

Definition 7 (Interaction Filter)**.** *Let $\eta(D)$ be the interaction predicate over D. An interactions filter, ϕ, is a function that returns all subsets of D_{group} that satisfy the interaction condition, i.e., $\phi(D_{group}, \eta(D)) = \{D \in D_{group} \mid \eta(D) = True\}$.*

For the co-location interactions we define the condition:

$$\eta_{co\text{-}locate}(D) = \forall d, d' \in D \ \{(d'.s < d.c \ \wedge \ d'.c > d.s) \ \wedge \ d.l = d'.l\}. \quad (1)$$

with which we operate the filtering function on D_{group}, and obtain D_{filter}, filtered subsets of D_{group} that corresponds to an interaction. As a last step, every set D_{filter} needs to be converted into a set of interactions that correspond to Definition 4. For this last step we define an interaction constructor function.

Definition 8 (Interaction Constructor)**.** *An interaction constructor is a mapping, ξ, which receives $D \in D_{filter}$, and returns a set of interaction tuples $\{(E, l, s, c)\}$.*

The set of interactions I that is mined from D_{agg} (through selection, grouping, and filtering) is defined as $I = \{\bigcup_{D \in D_{filter}} \xi(D)\}$. Note that ξ is a set to set mapping, since every set in D_{filter} may correspond to several interactions. For example, if $D \in D_{filter}$ contains co-location of doctor Bob, patient Anna and nurse Jenna. When doctor Bob enters, he may start a new interaction with the two entities (e.g., examination by physician and nurse). However, it may be that the doctor's entry is not related to the ongoing procedure. Since we need to consider all possible options, a single co-location set $D \in D_{filter}$ may correspond to several possible interactions.

To demonstrate the mining of the co-location interactions set $I_{co\text{-}locate}$ we return to Fig. 3 that corresponds to the RO log in Table 1, and focus on the interaction of the Exam activity (the rightmost interaction). We observe three time intervals in the RO log corresponding to nurse Bob, patient Anna and doctor Jenna, interacting in an examination room. According to our definition, while nurse Bob is alone in the room (at the beginning) and while doctor Jenna is alone in the room (at the end), there are no ongoing interactions. When patient Anna first enters the room, a possible interaction between nurse Bob and patient Anna is recorded. When Jenna enters the room, an interaction may start between either Bob and Jenna, or Anna and Jenna, or all three together; all

three options are considered to be part of $I_{co\text{-}locate}$. Furthermore, the interaction between nurse Bob and patient Anna may continue, as Jenna could be visiting the room unrelated to the ongoing activity.

Formally, let D_{filter} be a set of the filtered sets that correspond to the co-location interaction. Then, the interaction set $I_{co\text{-}locate} = \{(E, l, s, c)\}$ is mined by applying an interaction condition $\xi_{co\text{-}location}$ to every $D \in D_{filter}$ with the condition being

$$\xi_{co\text{-}location}(D) = \{(E = \bigcup_{d \in D'} d.b, l = D'.l, s = \max_{d \in D'} d.s, c = \min_{d \in D'} d.c) \mid D' \subseteq D\}, \tag{2}$$

and $D'.l$ being the shared location for every set $D' \in D$.

In the remainder of this work, we focus only on co-location interactions, since they are key indicators for service-oriented activities that require the presence of cases and resources. However, considering only the co-location interaction has the disadvantage of missing activities that do not require more than a single entity, such as a nurse examining blood results. To capture the latter, complementary interactions can be applied. For example, filtering interactions that involve special locations (e.g. blood laboratory), and certain types of entities (e.g. nurses).

4 Optimal Mapping of Interactions to Activities

In this part, we formulate an optimal matching problem (OMP) between the interaction set I and the activity set A. Specifically, we consider the interaction set $I_{co\text{-}locate}$, which we obtain by the mining technique proposed in Sect. 3.1. First, we write the matching problem as an Integer Linear Program (ILP). Then, we define process knowledge and demonstrate its encoding into the ILP. The solution to the OMP as an ILP results in a mapping between the RO log and the AD log, which is then easily transformed into an event log, hence solving the ROAD problem.

4.1 The Optimal Matching Problem

Let I be the interaction set (e.g., $I_{co\text{-}locate}$) and A be the set of activity labels. In this paper, we make a simplifying assumption that an interaction contains exactly one case. To ground the notion of a case in the process log we assume the existence of a case function $\tau : 2^B \to 2^B$ that returns the *case entities* (e.g., patients) from a set of entities E. For the matching problem, we consider only interactions with a single case:

$$I_c = \{i \in I \mid \exists b \in i.E \ (\tau(i.E) = \{b\}))\}. \tag{3}$$

Now, we turn to formulate the OMP problem as an Integer Linear Program, which consists of binary decision variables, a score function, and the constraints matrix [3].

Let $x_{i,a} \in \{0,1\}, i \in I_c, a \in A$ be binary decision variables that are assigned with the value 1 if interaction $i \in I_c$ is matched to activity label $a \in A$. Let $\mathbf{x}$ denote the vector of $x_{i,a}$, $g(\mathbf{x})$ be a linear score function, and B denote the constraint matrix of size $|I_c| \times |A|$. The ILP formulation of the matching problem is:

$$\underset{\mathbf{x}}{\text{maximize}}\ g(\mathbf{x}) \quad \text{subject to} \quad B^{\intercal}\mathbf{x} \leq 0 \tag{4}$$

The derivation of the AD log from the solution to the ILP is done by creating a tuple in the AD log for every $x_{i,a} = 1$ such that $(b, a, E, l, s, c) = (\tau(i.E), a, i.E \setminus \tau(i.E), i.l, i.s, i.c)$.

4.2 Encoding Process Knowledge into the ILP

We are now ready to demonstrate the instantiation of the ILP problem via an encoding of given process knowledge. We consider three types of process knowledge: (1) interaction knowledge (e.g., a doctor cannot be involved in two examinations simultaneously), (2) activity knowledge (e.g., the distributions of the infusion activity) and (3) behavioral knowledge (e.g., precedence constraints among activities for patients). These knowledge types can be derived from process-related documents, interviews with process experts, appointment books, and process models.

Interaction Knowledge – Pruning Alternative Co-locations: Interaction knowledge considers the activity dynamics of the underlying process. For example, one may decide that hallway interactions between patients are not process interactions and should not be considered as activity candidates. Such knowledge may reduce the size of I_c, therefore improving the performance of the ILP solver. To demonstrate how interaction knowledge may assist in pruning alternative interactions from I_c, we make the following assumptions:

A1 : An entity cannot be involved in two interactions at the same time.
A2 : An activity instance corresponds to at most a single interaction in I_c, and every interaction stems from at most a single activity instance.

From A1 we get that interactions that overlap in time and intersect in the set of involved entities cannot co-exist in the mapping. Clearly, this may not be the case in every business process: sometimes resources may participate in interactions with two case entities simultaneously. The second assumption, A2, prevents from mapping interactions to more than a single activity. Here, we assume that two interactions cannot stem from a single activity instance. This may not hold in processes where activities have a complex life cycle with interrupts.

We start by encoding assumption A1 into the ILP. As a preprocessing step, we mine an exclusion relation $I_X \subseteq I_c \times I_c$ from I_c such that $(i_1, i_2) \in I_X \iff i_1.E \cap i_2.E \neq \emptyset$, and i_1, i_2 overlap in time. For every pair of these interactions we add a constraint to the ILP allowing at most one interaction to be mapped to an activity, otherwise their joint mapping would be inconsistent:

$$\forall (i_1, i_2), \in I_X : \sum_{a \in A} x_{i_1,a} + \sum_{a \in A} x_{i_2,a} \leq 1. \tag{5}$$

Next, we demonstrate the encoding of assumption A2 into B. The interaction set I_c can be partitioned according to case entities, since every $i \in I_c$ contains exactly one case entity $\tau(i)$. Let I_c^j be the set of interactions that corresponds to case entity $j \in J$, with J being the set of case entities. Then, we write the constraints for A2 in a way that allows an interaction to be mapped to at most a single activity label:

$$\forall i \in I_c : \sum_{a \in A} x_{i,a} \leq 1; \quad \forall j \in J, \forall a \in A : \sum_{i \in I_c^j} x_{i,a} \leq 1. \tag{6}$$

Further pruning on the interaction level is possible. For example, one may consider interactions with durations only above a certain threshold, and allow for only a single interaction per each location at a time.

Activity Knowledge – Durations, Resources, and Locations: Here, we consider specific types of activity knowledge, namely durations, possible resource assignments, and possible locations of activity executions. Such knowledge is often available in historical patient records, appointment books and models. We take a stochastic perspective on activity knowledge, assuming components to be random. Formally, we define the activity knowledge components as random variables and specify their corresponding distribution functions. For every activity $a \in A$, let D_a, E_a, and L_a be the random durations, random sets of assigned resources, and random locations, respectively. We denote by f_{D_a}, f_{E_a}, and f_{L_a} the probability distribution functions (PDFs) for these random variables, respectively. Therefore, $f_{D_a}(d) = Pr\{D_a = d\}, f_{E_a}(E) = Pr\{E_a = E\}$, and $f_{L_a}(l) = Pr\{L_a = l\}$.

We assume that activity knowledge is shared among all instances. However, our model is general enough to consider instance-level knowledge (e.g., the distribution of patient Anna's infusion duration from her past visits). Below, we demonstrate the encoding of activity knowledge into the ILP score function $g(\mathbf{x})$. We attach a reward coefficient to each of the decision variable $x_{i,a}$, which is defined as:

$$w_{i,a} = Pr\{x_{i,a} = 1 \mid i = (E, l, s, c)\}. \tag{7}$$

The coefficient $w_{i,a}$ can be interpreted as the posterior probability that interaction i is mapped to activity a given that i has values (E, l, s, c). By applying Bayes theorem:

$$w_{i,a} = \frac{\pi_a Pr\{i = (E, l, s, c) \mid x_{i,a} = 1\}}{Pr\{i = (E, l, s, c)\}}, \tag{8}$$

with $\pi_a = Pr\{x_{i,a} = 1\}$ being the prior that any interaction maps to activity a, $Pr\{i = (E, l, s, c) \mid x_{i,a} = 1\}$ being the likelihood of interaction $i = (E, l, s, c)$ if it comes from activity a, and $Pr\{i = (E, l, s, c)\}$ being the probability to get the values (E, l, s, c) by randomly selecting an interaction from I_c. The denominator in Eq. 8 is a scaling factor that does not depend on the $x_{i,a}$ and can therefore be excluded from the score function. Further, we assume that the three knowledge components, D_a, E_a, and L_a are independent, which allows us to write the

following multiplicative form for $w_{i,a}$:

$$w_{i,a} = \pi_a Pr\{i.E, i.l, i.c - i.s \mid x_{i,a} = 1\} = \pi_a f_{D_a}(i.c - i.s) f_{E_a}(i.E) f_{L_a}(i.l). \quad (9)$$

Thus, the score function can be written as:

$$g(\mathbf{x}) = \sum_{i \in I_c} \sum_{a \in A} w_{i,a} x_{i,a} = \sum_{i \in I_c} \sum_{a \in A} \pi_a f_{D_a}(i.c - i.s) f_{E_a}(i.E) f_{L_a}(i.l) x_{i,a}. \quad (10)$$

The assumption that we make in Eq. 10 is that the reward is additive for matching i to a, and is linear in the likelihood of i to be an interaction coming from activity a according to durations, resources and locations.

Table 2. Left-hand side – soft encoding; right-hand side – hard encoding.

Soft encoding	Hard encoding
$\forall (a,b) \in \prec_P, \forall i \in I_c:$	$\forall (a,b) \in \prec_P, \forall i \in I_c:$
$z_{i,a,b} \leq 1 - x_{i,b} + y_{i,a}$	$x_{i,b} \leq y_{i,a}$
$z_{i,a,b} \geq 1 - x_{i,b}$	
$z_{i,a,b} \geq y_{i,a}$	
$z_{i,a,b} \in \{0,1\}$	

Behavioral Knowledge – Precedence Order: Behavioral knowledge is a relation between activities in A, which can be obtained from various sources, *e.g.*, by computing behavioral profiles from a given process model [4]. In DayHospital, one may use the schedule of patients to derive precedence constraints between activities. Here, we demonstrate the encoding of a precedence order between activities. Let $\prec_P \subseteq A \times A$ be the precedence relation, where $a \prec_P b$ if b's execution implies that $c_a \leq s_b$, with c_a being the completion time of a and s_b being the start time of b. Further, we assume that loops do not exist, and if an activity a is repeated it corresponds to a new activity label, a'.

As a preliminary phase to encoding precedence knowledge, let $P : I_c \rightarrow 2^{I_c}$ be the precedence function for interactions in I_c such that $P(i)$ returns the set all interactions that precede i, and have the same case entity (e.g. the same patient identifier): $\{j \in I_c \mid \tau(i) = \tau(j), i.c \leq j.s\}$. Let $y_{i,a} = \sum_{j \in P(i)} x_{j,a}$ denote the sum of mappings between $j \in I_c$ in precedence to $i \in I_c$ that map to a. $y_{i,a}$ indicates if an interaction that precedes i was mapped to a.

To encode $\prec_P$ into the ILP problem, we have the choice of adding hard constraints (into B), thus preventing violations in precedence order, or assigning smaller rewards for mappings that violate $\prec_P$. Table 2 summarizes the formulation of the two options. For hard encoding of the precedence constraints we add the constraints on the right-hand side of Table 2 into B. For soft encoding, we add additional variables $z_{i,a,b} \in \{0,1\}$ to the ILP and add constrains into B as stated on the left-hand side of Table 2. These constraints are equivalent to the

logical predicate $x_{i,b} \rightarrow y_{i,a}$. Having defined $z_{i,a,b}$, we add these variables and their corresponding rewards into the score function of the ILP program. The size of the reward for a matching that does not violate precedence constraints is user-defined, and is set by default to the median of $w_{i,a}$, as defined in Eq. 7. These default values are used to scale precedence violation weights in correspondence with activity knowledge weights.

5 Evaluation

In this section, we present the evaluation of our ROAD solution via controlled experiments. Specifically, we introduce the experimental setting, present results, and discuss the main factors that influence the accuracy of our approach.

Experimental Setting. We implemented our solution to the ROAD problem in the ProM framework.[1] The design of our experiment is depicted in Fig. 4 and consists of five steps. In step 1, we generate a simulation ready stochastic Petri net (SPN) model based on knowledge from the healthcare process described in Example 1, and the process model in Fig. 1. Note that model parameters vary across three different scenarios that we use to test the sensitivity of our solution. In step 2, the SPN is simulated to create the ground truth AD log, denoted L. In step 3, the AD log is converted (by label removals) into the RO log, which contains time intervals involving sets of entities in different locations. In step 4, we use process knowledge and apply the ROAD approach to the RO log, which results in a reconstructed AD log, denoted $\hat{L}$. Step 5 computes a similarity measure that quantifies the difference between L and $\hat{L}$.

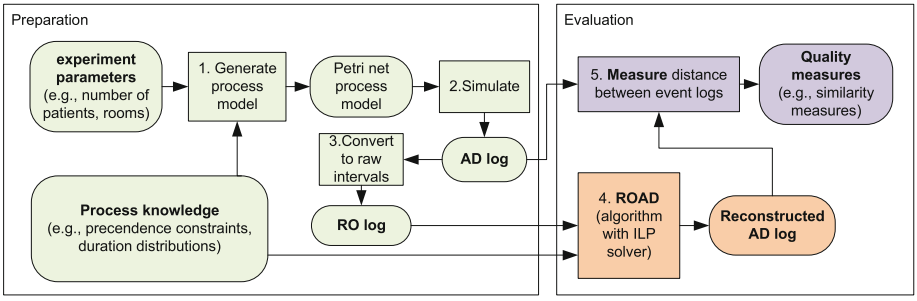


Fig. 4. Evaluation setting to test accuracy of the ROAD approach.

We use a multi-dimensional similarity measure that consists of four quantifiers. For reconstructing trace activity labels (including their order) we use the complementary of average Levenshtein distance measure between two traces [5]. For duration similarity we consider the complementary of the symmetric mean

[1] See `StochasticNet` package. http://www.promtools.org.

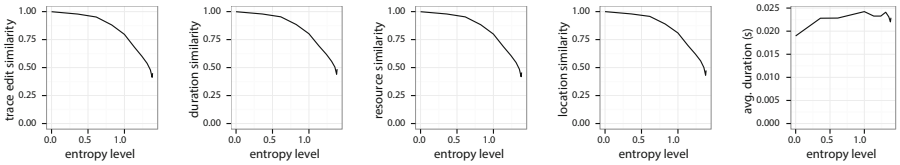
absolute percentage error (sMAPE) [6]. For comparing two resource sets we use the average Jaccard similarity, and for location similarity we use the average of an indicator that is set to 1 if the location was reconstructed correctly (and 0 otherwise).

To test the accuracy and sensitivity of our ROAD solution, we consider the following three scenarios. In *scenario 1*, we alter the level of entropy in process knowledge for resource assignments and activity locations, namely E_a and L_a. Specifically, we change the probability distributions, f_{E_a}, f_{L_a}, from deterministic (no entropy), to uniform distributions (maximal entropy). We expect that deterministic values will result in accurate discrimination between activities, while for maximal entropy, the performance of our solution will deteriorate. For example, it is more difficult to reconstruct activity instances for locations that are used for multiple activities, as opposed to locations that support a single activity.

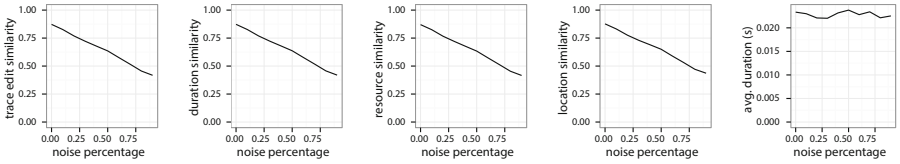
In *scenario 2*, we introduce noise in the form of deviations in the execution of the process with respect to the existing process knowledge. For example, we insert swaps between activity instances in the simulated AD log such that precedence constraints are violated. Further, we allow for changes in location and assigned resource sets. We hypothesize that an increase in noise will cause a reduction in similarity measures.

Last, in *scenario 3*, we increase the number of activities that can occur in the same time and place. This, by definition of the co-location interactions, results in an exponential increase in the size the interaction set $I_{co\text{-}locate}$. We hypothesize that this exponential increase will result in an accuracy reduction, as well as in run-time deterioration. Note that in scenarios 2 and 3, the entropy level in the process knowledge corresponds to the realistic values coming from the real-life process described in Example 1.

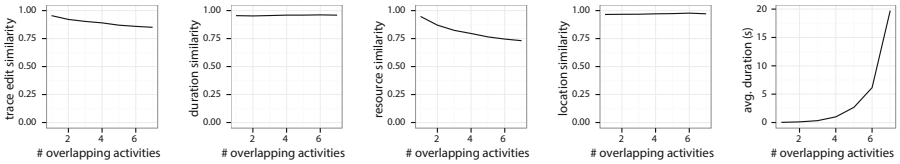
Results. The results of our evaluation for the three scenarios are presented in Fig. 5a, b, and c, respectively. The vertical axes in these figures correspond to the four aforementioned similarity measures, and to the run-time performance of our approach (in seconds). The horizontal axes for the three scenarios correspond to the level of entropy in the process knowledge (Fig. 5a), the noise percentage (Fig. 5b), and the maximal number of overlapping activities per location (Fig. 5c). We omit the formal definition of entropy level and noise percentage due to space limitations. Figure 5a shows a steep decline in all four similarity measures as the level of entropy approaches its maximal value; the run-time shows a negligible increase across entropy levels. As for scenario 2 (Fig. 5b) we observe a linear decline in accuracy, i.e., the error is proportional to the noise inserted. For scenario 3 (Fig. 5c) a mild decrease in accuracy is evident, while we observe an exponential growth in run-time. It is worth noting that for scenarios 1 & 2, all similarity measures display an almost identical behavior, which means that any aggregated similarity measure would demonstrate the same pattern. For scenario 3, while we see a mild decrease for all measures, the shape of the decrease vary slightly.



(a) Similarity with respect to increasing entropy in the mapping to activities.



(b) Similarity with respect to increasing noise reflecting deviations from the process knowledge.



(c) Similarity with respect to increasing overlap in activities per location.

Fig. 5. Accuracy results assessing the ROAD solution.

Discussion. Our evaluation shows that the quality of process knowledge has most influence on the accuracy of the ROAD solution. Specifically, process knowledge becomes less informative as entropy increases, and cannot be applied to reconstruct the AD log. The second most relevant source for inaccuracies stems from deviations in process knowledge (i.e., the noise factor). Further, our definition for co-locating interactions leads to an exponential growth in the size of the considered interaction set, and may pose computational limitations. However, in real-life processes, we seldom observe multiple overlapping activities in a single location.

To conclude, the ability to go from RO to AD using our approach depends on both the informativeness of the process knowledge (for higher accuracy), and the definition of interactions (for lower run-time complexity).

6 Related Work

Our research is most closely related to automatic process discovery, process alignment, and activity recognition. Bridging the abstraction gap between raw events and activity data has been a subject of several recent works in process mining, c.f. [7] and the references within. In [7,8], a semi-automated approach was

proposed where process knowledge is used to match raw events to process activities; the approach is extended to use constraint programming in [9]. Our work generalizes these approaches by optimally mapping raw data to activities, and by considering further dimensions (e.g. time, resources). In [10], classification and regression techniques are used to create process views from low-level multi-dimensional data, without assuming the existence of pre-defined activity labels. Their approach uses logs with process related raw events. Other approaches in process mining for connecting low-level events with activities include clustering [11], Expectation-Maximization based sequence mining [12]. These techniques focus on structural and behavioral aspects of the control-flow perspective, which our work extends towards a multi-perspective approach. None of the aforementioned works in process mining consider tracking data, but rather event logs that come from information systems executing the processes.

Our approach is also related to a multi-perspective conformance checking and alignment of event logs to process models, c.f. [13] and the references within. We propose an optimal matching between interactions and activities while considering event logs that are not semantically related to activities. In order to reduce the search space, we adopt techniques from process matching, where processes are matched according to structural and behavioral similarities [14]. Our approach includes behavioral matching according to precedence constraints, in the spirit of [4].

In our solution, we are also inspired by techniques for sensor-based *activity and event recognition*. The former is a well-established task in Artificial Intelligence [15]. Methods for activity recognition include two main approaches: data-driven and knowledge-driven activity recognition. Event recognition, a related task, is a well-studied problem in the field of complex event processing [16]. Similarly to our interaction mining technique, the idea behind these works is to use logical predicates to filter events. However, state-of-the-art event and activity recognition techniques do not assume a process perspective. In our work we take the knowledge-driven approach to activity recognition, based on logical predicates, in the spirit of [16,17], while introducing the context of processes, which in turn creates dependencies between activities. To conclude related work, we narrow the scope to literature on mining location data. A methodology for clustering RFID trajectories to reconstruct entity paths was proposed in [18]. Moreover, a probabilistic model for workflow discovery from RTLS data was applied in [19]. In contrast to our solution, the former work disregards the process perspective, while the latter assumes that locations correspond to activities in a one-to-one fashion.

7 Conclusion

In this work, we provided a transformation of sensor data (e.g. Real-Time Locating System data) into standard event logs, to enable the application of process mining techniques to raw location recordings. The transformation was based on the notion of interactions, which is an intermediate knowledge layer that bridges

between raw sensor data, and process instances. After mining interactions from the raw data, we solved a matching problem that is based on process knowledge. The solution to the problem finds optimal correspondences between the interactions and activity labels, and creates activity instances, which comprise the target event log. We evaluated the approach with controlled experiments by using simulated event logs. The experiments show that the accuracy of our technique depends on the informativeness of process knowledge, while the complexity of the technique depends on the number of possible interactions.

In future work, we aim at a feature complete encoding of process models into the optimal matching setting. This requires a process model at hand, which can be obtained via process discovery. Such encoding would enable further automation of our ROAD solution. Moreover, we would like to test our solution on real-world processes that emit sensor data by cross-validating the resulting event log against information that comes from process-aware systems that accompany and execute the process.

Acknowledgment. This work was supported by the EU project SERAMIS (612052).

References

1. van der Aalst, W.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
2. Etzion, O., Niblett, P.: Event Processing in Action. Manning Publications Co., Greenwich (2010)
3. Schrijver, A.: Theory of Linear and Integer Programming. Wiley, Chichester (1998)
4. Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 319–334. Springer, Heidelberg (2012)
5. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM **21**(1), 168–173 (1974)
6. Hyndman, R.J., Koehler, A.B.: Another look at measures of forecast accuracy. Int. J. Forecast. **22**(4), 679–688 (2006)
7. Baier, T., Mendling, J., Weske, M.: Bridging abstraction layers in process mining. Inf. Syst. **46**, 123–139 (2014)
8. Baier, T., Mendling, J.: Bridging abstraction layers in process mining by automated matching of events and activities. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 17–32. Springer, Heidelberg (2013)
9. Baier, T., Rogge-Solti, A., Weske, M., Mendling, J.: Matching of events and activities - an approach based on constraint satisfaction. In: Frank, U., Loucopoulos, P., Pastor, Ó., Petrounias, I. (eds.) PoEM 2014. LNBIP, vol. 197, pp. 58–72. Springer, Heidelberg (2014)
10. Folino, F., Guarascio, M., Pontieri, L.: Mining predictive process models out of low-level multidimensional logs. In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 533–547. Springer, Heidelberg (2014)

11. Günther, C.W., Rozinat, A., van der Aalst, W.M.P.: Activity mining by global trace segmentation. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 128–139. Springer, Heidelberg (2010)
12. Ferreira, D.R., Szimanski, F., Ralha, C.G.: Mining the low-level behaviour of agents in high-level business processes. Int. J. Bus. Process Integr. Manag. **6**(2), 146–166 (2013)
13. Mannhardt, F., de Leoni, M., Reijers, H., van der Aalst, W.: Balanced multi-perspective checking of process conformance. Computing **98**(4), 407–437 (2016)
14. Dijkman, R., Dumas, M., Van Dongen, B., Käärik, R., Mendling, J.: Similarity of business process models: metrics and evaluation. Inf. Syst. **36**(2), 498–516 (2011)
15. Chen, L., Hoey, J., Nugent, C.D., Cook, D.J., Yu, Z.: Sensor-based activity recognition. IEEE Trans. Syst. Man Cybern. B Cybern. **42**(6), 790–808 (2012)
16. Artikis, A., Skarlatidis, A., Portet, F., Paliouras, G.: Logic-based event recognition. Knowl. Eng. Rev. **27**(04), 469–506 (2012)
17. Azkune, G., Almeida, A., López-de Ipiña, D., Chen, L.: Extending knowledge-driven activity models through data-driven learning techniques. Expert Syst. Appl. **42**(6), 3115–3128 (2015)
18. Han, Y., Tucker, C.S., Simpson, T.W., Davidson, E.: A data mining trajectory clustering methodology for modeling indoor design space utilization. In: ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers V03BT03A017–V03BT03A028 (2013)
19. Liu, C., Ge, Y., Xiong, H., Xiao, K., Geng, W., Perkins, M.: Proactive workflow modeling by stochastic processes with application to healthcare operation and management. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2014, pp. 1593–1602. ACM, New York (2014)