Situation Mining: Event Pattern Mining for Situation Model Induction

Andrea Salfinger
Department of Cooperative Information Systems
Johannes Kepler University Linz
Altenberger Strasse 69
4040 Linz, Austria
andrea.salfinger@cis.jku.at

***Abstract*—Computational situation assessment (SA) systems support human control center operators in *situation monitoring*, i.e., detecting and tracking relevant object and event constellations in their observed environment. SA systems frequently employ deductive reasoning techniques implemented in Complex Event Processing or rule engines to solve this real-time pattern recognition problem, by matching data sensed from the monitored environment against templates for those situations, characterizing the event patterns of interest. Hence, they require explicitly formalizing the sought-after types of situations, demanding human domain experts to conceptually model their cognitive situation hypotheses, which represents a time-consuming and non-trivial task. To overcome this *situation knowledge acquisition bottleneck*, we therefore propose an approach for *inductive situation modeling* to leverage existing data sets of recorded situations: We contribute a dedicated *situation mining* algorithm, which bootstraps situation model acquisition by automatically mining behavioral models of situations, so-called situation evolution models, from already observed situation instances. The feasibility of our approach is examined on a case study from the domain of road traffic incident management, to demonstrate how it turns previously implicit knowledge hidden in the situation instances into explicit situation knowledge that can be inspected and queried for *situation analytics*, and sketch how the derived situation evolution models can be used within a Model-Driven Engineering framework to directly generate the corresponding rule code for automated *situation assessment*.**

***Keywords*—event pattern mining, knowledge acquisition**

I. Introduction

Motivation. Computational situation assessment (SA) systems support human control center operators in *situation monitoring*, i.e., detecting and tracking relevant object and event constellations in their observed environment. SA systems frequently employ deductive reasoning techniques implemented in Complex Event Processing (CEP) or rule engines to solve this real-time pattern recognition problem, by matching data sensed from the monitored environment against the situation patterns of interest, for instance as demonstrated for the domains of logistics [1], sea surveillance [2], road traffic control [3], personalized situation alerts [4], environmental monitoring [5], fleet management [6], and agriculture [7].

Situation Model Acquisition. Hence, these systems require characterizing the sought-after types of situations in the form of logical specifications of their constituting event constellation, so that these can be translated into corresponding situation detection rules that will trigger an alert whenever this event constellation is observed. Thus, human domain experts need to conceptually model their cognitive situation hypotheses, which represents a time-consuming and non-trivial task. Furthermore, domain experts' knowledge may be tacit or incomplete, leading to the *situation knowledge acquisition bottleneck* [8] familiar from expert systems research. Explicitly modeling their situation knowledge is perceived cumbersome and error-prone, as human domain experts usually intuitively recognize actual situation instances and their constituting elements during their control center monitoring routines.

Approach. Therefore, we propose to bootstrap *situation model acquisition* by automatically mining situation models from previously recorded situation instances, using the meta-model employed by the human modeler as hypothesis language to generate models in the human's modeling language, which thus can be revised by the human modeler. This contrasts our approach to related inductive approaches (i.e., approaches that generalize models from instance data) basing on sub-symbolic machine learning techniques, which employ different meta-models, termed pattern or hypothesis languages (or model classes) [9] in machine learning terminology, than the human modeler. Our specific goal will be to reconstruct behavioral models of situations [10], so-called situation evolution models [11], from already observed situation instances, using a dedicated Situation Evolution Meta-Model ($SE\mathcal{M}$) [11] as pattern language, which represents a situation's different evolutionary states within a state-transition-system. Hence, the joint evolution of a complex set of monitored objects is discretized into its different relational states, to provide a concise representation of the overall observed, relational evolution patterns.

Contributions. For realizing the proposed inductive situation modeling approach, we contribute a dedicated *situation mining* algorithm. We demonstrate the benefits of this inductive approach on a case study from the domain of road traffic incident management, to illustrate how it allows to turn previously implicit knowledge hidden in the situation instances data into explicit situation knowledge that can be inspected and queried for *situation analytics*, and sketch how the mined situation evolution models can be used within a Model-Driven Engineering (MDE) framework to directly generate the implementation code for automated *situation assessment*.

Structure of the Paper. In the next section, we provide the relevant background on the conducted case study, before formally specifying our problem in Section III. Section IV introduces the developed approach, which is evaluated on our case study

in Section V. Further applications of situation mining are outlined in Section VI. Section VII compares related work on situation modeling and mining. Finally, section VIII draws conclusions and presents ideas for future work.

II. Motivation and Background

In the following, we will introduce the motivational background of our case study in the domain of road traffic incident management. The analyzed road traffic incident records have been obtained from ASFINAG[1], the company that maintains and operates Austria's highway network, and are conceptually based upon the "Data Exchange for Traffic Management and Travel Information" standard, DATEX II[2]. DATEX II has been developed by the European Committee for Standardization as a common specification for exchanging traffic information between traffic management centers, traffic service providers, traffic operators and media partners in XML-format. It describes road traffic incident data in form of *situations*, whereby `Situation` is defined as "An identifiable instance of a traffic/travel situation comprising one or more traffic/travel circumstances which are linked by one or more causal relationships. Each traffic/travel circumstance is represented by a Situation Record."[3]. Thus, the complex type `Situation` groups causally related road traffic objects and events, referred to as `SituationRecords`. Since the described situation may evolve over time, this element is versioned and has unique identifier. Naturally, also the `SituationRecord` element may evolve over time, which is thus also versioned and has unique identifier. In the following, we will use the term `Object` instead of `SituationRecord`, in order to comply with the domain agnostic and thus more widely adopted JDL data fusion terminology, which defines a situation as *a set of objects in relations* [12].

The comprised `Objects` can be of three main categories, notably *traffic elements* (events not planned by the traffic operator, which are affecting, or have the potential to affect traffic flow, including events planned by external organizations such as exhibitions, sports events etc.)[4], *operator actions* (actions that a traffic operator can decide to implement to prevent or help correct dangerous or poor driving conditions, including maintenance of the road infrastructure)[5], as well as *non-road event information* (information about an event which is not on the road, but which may influence the behavior of drivers and hence the characteristics of the traffic flow)[6].

Our data set has been compiled from live recording of such real-time road traffic `Situation` publications: In roughly three-minute update intervals, ASFINAG published a list of currently observed road traffic `Objects`, which has been stored in a CSV file. Each row in this CSV file holds the information on the *current state* of an observed traffic `Object`, including its encompassing `Situation's` ID, the type of observed event, as characterized by DATEX DOB (object code defining the type of event) and PHR codes (phrase code defining the type of event) [13], its begin time, location on the road network, unique identifier, and the message shown to approaching motorists on Variable (Traffic) Message Signs (VMS). This incident information originates from ASFINAG's roadworks management system and the public road traffic information service provided by the Austrian radio station Hitradio Ö3[7], which receives notifications about on-going traffic situations from motorists calling its traffic newsroom operators. Table I outlines the basic characteristics of the analyzed data set. We note that situations are only implicitly given in this data set, as each traffic object specifies its encompassing situation's ID.

Message Publication Time Span	2014-12-27 23:59 - 2015-06-14 00:00
# Situations ($\mathcal{S}^{\mathcal{I}}$)	1887
# (Traffic) Objects ($\mathcal{O}^{\mathcal{I}}$)	2771
# Object States (Updates) ($o_t^{\mathcal{I}}$)	4445
# Object Types (DATEX PHR Codes) ($\mathcal{O}^{\mathcal{T}}$)	110

Table I: Data Set Characteristics

Concluding, DATEX II provides a generic data structure for representing *situation instances*, i.e., supports a grouping of objects to form situations, but lacks any more specific "typed" information on the different kinds of situation we might observe. However, we might use these instances to learn what different "types" of situations we have been observing, i.e., derive descriptive models characterizing the observed situation instances.

Thus, in the following sections, we will try to answer the following questions:

- Can we reconstruct abstract models summarizing the observed situations' evolutions?
- Can we compile these models to an implementation allowing for automatically detecting and tracking such situation instances?

III. Problem Statement

Based on this motivational scenario, in the following, we will formalize our problem.

A. Requirements

The input to our *situation mining* approach is a set of situation instances $\mathcal{S}^{\mathcal{I}}$, i.e., each recorded situation has been labeled with a unique ID ($\mathcal{I}$). Each situation comprises one or multiple objects. Situation instances might have been created by human operators responsible for their situation life-cycle management, who handled the corresponding case. Our goal will be to derive abstract specifications for these situation instances, which we term *Situation Evolution Types* ($SE^{\mathcal{T}}$s) –

[1] `www.asfinag.at`

[2] `www.datex2.eu`

[3] `http://d2docs.ndwcloud.nu/level2user/level2PayloadPublication.html`

[4] different types of abnormal traffic, activity, accident, conditions, equipment or system fault, obstruction

[5] different types of roadworks, sign setting, network management, roadside assistance

[6] different types of transit information, information about other transport means, e.g., cancellation, or delay, on a tram, train, plane, etc. journey; service disruption (rest area closed, no diesel etc.), road service disruption (no patrol, emergency call out of order, etc.)

[7] `https://oe3.orf.at/programm/stories/verkehrsredaktionstelltsichvor/`

since these lift the observed instance-level information (of an untyped and thus "arbitrary" situation instance $\mathcal{S}^{\mathcal{I}}$) to type-level information characterizing its composition in an abstract fashion. Note that in the following, we will use super-scripts to notationally distinguish instance-level ($^{\mathcal{I}}$) from type-level ($^{\mathcal{T}}$) information.

We further require that also each encompassed object record is identified by a unique ID, and of a particular object type ($\mathcal{O}^{\mathcal{T}}$). Objects might evolve, i.e., we receive updates on this object - in this case, we observe a new object state record with the same ID but different begin time, i.e., we can reconstruct this object's evolution as a sequence of its object states, which are temporally ordered by their begin times. An object's type $\mathcal{O}^{\mathcal{T}}$ might change throughout its evolution, for instance, we might observe the following change of $\mathcal{O}^{\mathcal{T}}s$ between two object states: `LS1` $\rightarrow$ `LS2`, meaning that an object starting in an object state of type LS1 ("stationary traffic") evolved to LS2 ("queuing traffic"), or `ACI` $\rightarrow$ `ACX`, meaning that an object starting as type "accident" evolved to a state of type "accident cleared".

Formalizing these requirements, we denote the evolution of an *object* $\mathcal{O}^{\mathcal{I}}$ by a sequence of observations[8] of its different object states $o_t^{\mathcal{I}}$:

$$\mathcal{O}^{\mathcal{I}} :=< o_{t1}^{\mathcal{I}}, \ldots, o_{tn}^{\mathcal{I}} >, \tag{1}$$

where n denotes the number of observed object states (starting at time instants $t = t1$ to tn), i.e., updates.

Analogously to formulating object evolution, a situation's evolution over its entire situation life-cycle would be formalized as

$$\mathcal{S}^{\mathcal{I}} :=< s_{t1}^{\mathcal{I}}, \ldots, s_{tn}^{\mathcal{I}} >, \tag{2}$$

whereby $\mathcal{S}^{\mathcal{I}}$ denotes a particular *situation instance*, which can be expressed as a sequence of situation state updates, $s_t^{\mathcal{I}}$, which we term *situation state instances*. Each situation state instance $s_t^{\mathcal{I}}$ simply represents a container for the set of k object states composing the situation state:

$$s_t^{\mathcal{I}} := \{o_t^i\}, \quad i \in \{1, \ldots, k\} \tag{3}$$

The *cardinality* of a situation state instance $s_t^{\mathcal{I}}$ thus is the size of its object state set, i.e., $\|s_t^{\mathcal{I}}\| := \|\{o_t^i\}\| = k$. Concluding, a situation instance simply represents a container for a changing set of related object states. Updates of a situation (i.e., a change of its situation state) correspond to updates of its contained objects.

B. Prior Work – the Meta-Model

To automatically detect and track such situation instances $\mathcal{S}^{\mathcal{I}}$ within object observations $\{o\}$ sensed from the monitored environment, deductive reasoning systems require abstract descriptions on the "type-level", which generalize the common patterns underlying the observed instances such that these can be translated to corresponding situation detection rules. In previous work, we have therefore proposed a conceptual meta-model, the *Situation Evolution Meta-Model* ($SE\mathcal{M}$) [11], for modeling different *types* of evolving situations, i.e., $SE^{\mathcal{T}}s$. By adopting a Model-Driven Engineering (MDE) approach, this meta-model supports the automated generation of the corresponding situation detection and tracking rules from the specified $SE^{\mathcal{T}}s$. The $SE\mathcal{M}$ models an evolving situation as a state transition system:

$$SE\mathcal{M} := (\mathbb{S}, \Lambda, \longrightarrow), \tag{4}$$

whereby $\mathbb{S}$ comprises a set of Situation State Types ($\mathcal{SS}^{\mathcal{T}}s$), which characterize a situation's different object-relational states, thereby corresponding to abstract descriptions for sought-after situation states $s_t^{\mathcal{I}}$. $\longrightarrow$ encodes the transitions between those states, i.e., the change from one relational state to a different one, thus corresponding to binary relations over $\mathbb{S}$, i.e., $\longrightarrow \subseteq \mathbb{S} \times \mathbb{S}$. Λ consists of the set of changes between two succeeding states, outlining what has changed within an evolution step. Concretely, $\mathcal{SS}^{\mathcal{T}}$s are defined as:

$$\mathcal{SS}^{\mathcal{T}} := (\Omega, \rho), \tag{5}$$

whereby Ω is a finite, non-empty set of *object references* ($\mathcal{O}^{\mathcal{R}}$). An object reference corresponds to an *object type* ($\mathcal{O}^{\mathcal{T}}$) referenced by an *alias*, such as *Wrong-way Driver* w or *Tunnel* t. The alias is required to distinguish between different objects of the same type in the reasoning process (e.g., different traffic jams j^1 and j^2), hence an $\mathcal{O}^{\mathcal{R}}$ can be considered as a variable (in the reasoning process, matching object states are bound to these variables). ρ is a set of n-ary relation types $\mathcal{R}^{\mathcal{T}}$s that need to hold between these $\mathcal{O}^{\mathcal{R}}$s, i.e., $\rho : \Omega^n \rightarrow \{true, false\}$ for the situation state to be true. Hence, an $\mathcal{SS}^{\mathcal{T}}$ is defined by a set of $\mathcal{O}^{\mathcal{R}}$s in specific $\mathcal{R}^{\mathcal{T}}$s, thereby characterizing a situation's different object-relational states, whereas a transition represents a change of the situation's relations, i.e., the real-world situation's evolution from one $\mathcal{SS}^{\mathcal{T}}$ to another. Thus, a $SE^{\mathcal{T}}$ spans all evolutionary states a real-world situation might potentially evolve through, i.e., encodes what could happen.

For automated situation detection, each $\mathcal{SS}^{\mathcal{T}}$ is compiled to a rule by a *model-to-text transformation* [14]. At runtime, object states $\{o_t\}$ matching a particular rule lead to the instantiation of a situation state instance $s_t^?$ of the corresponding $\mathcal{SS}^{\mathcal{T}}$. Situation tracking then performs a dedicated reasoning procedure to detect whether situation states $s_{t-1}^?$ and $s_t^?$ belong to the same situation instance $\mathcal{S}^{\mathcal{I}}$ (by performing reachability analysis on the $SE^{\mathcal{T}}$ and comparing the states' object compositions), and fusing these updates to a contiguous sequence of situation state

Name	Definition
$\mathcal{S}^{\mathcal{I}}$	Situation
$s_t^{\mathcal{I}}$	Situation State
$O^{\mathcal{I}}$	Object
$o_t^{\mathcal{I}}$	Object State
$\mathcal{O}^{\mathcal{T}}$	Object Type
$\mathcal{R}^{\mathcal{T}}$	Relation Type
$\mathcal{O}^{\mathcal{R}}$	Object Reference ($\mathcal{O}^{\mathcal{T}}$ + alias)
Ω	set of Object References $\{\mathcal{O}^{\mathcal{R}}\}$
ρ	set of Relation Types $\{\mathcal{R}^{\mathcal{T}}\}$
$\mathcal{SS}^{\mathcal{T}}$	Situation State Type
$SE^{\mathcal{T}}$	Situation Evolution Type
$SE\mathcal{M}$	Situation Evolution Model

Table II: Notations

[8]Notation: We denote sets by $\{\ldots\}$ and sequences by $< \ldots >$.

updates, representing the situation instance $\mathcal{S}^{\mathcal{I}}$. Thus, situation tracking essentially reconstructs a real-world situation instance's actual evolution by tracking its path through the $SE^{\mathcal{T}}$, i.e., represents a situation instance by the sequence of $\mathcal{SS}^{\mathcal{T}}$s it has evolved through. Hence, different real-world situation instances of the same $SE^{\mathcal{T}}$ may expose different $\mathcal{SS}^{\mathcal{T}}$ sequences. Thus, an $\mathcal{SS}^{\mathcal{T}}$ essentially provides a "type description" for the situation states s_t introduced in Eq. 3, whereas $SE^{\mathcal{T}}$s denote which transitions, i.e., evolutions, are possible.

C. Goal

In previous work [11], [15], we have assumed that human domain experts model these $SE^{\mathcal{T}}$s, which then allows tracking situation instances according to this model. Thus, we generate the situation instances from the models:

$$\underbrace{(\{SE^{\mathcal{T}}\}}_{\text{human-provided}} \underbrace{,\{o\}) \xrightarrow{\text{situation assessment}} \{\mathcal{S}^{\mathcal{I}}\}}_{\text{automated}}$$

In the present work, we take the opposite direction: We aim at inducing such $SE^{\mathcal{T}}$s from already observed, human-labeled situation instances $\mathcal{S}^{\mathcal{I}}$, thus, we generalize the situation models from the situation instances:

$$\underbrace{\{SE^{\mathcal{T}}\} \xleftarrow{\text{situation mining}}}_{\text{automated}} \underbrace{\{\mathcal{S}^{\mathcal{I}}\}}_{\text{human-provided}}$$

Both approaches use the same meta-model, the SEM, for instantiating concrete *models*, i.e., types, of evolving situations ($SE^{\mathcal{T}}$s). Thus, $SE^{\mathcal{T}}$s can be instantiated by the human modeler or automatically induced via situation mining, which also allows for interleaving human-driven and data-driven modeling. In the following, we will outline how we realize this mining approach, exemplified on our case study. We note that while evaluated on our previously described road traffic incident data conforming to the DATEX II standard, our approach in general only requires labeled situations (each situation instance must be identifiable by a unique ID), which are composed out of a set of evolving objects of distinct types, and is thus applicable to various evolving, object-relational problem domains. Table II summarizes our general notational conventions.

IV. Approach

Starting from our situation instances $\mathcal{S}^{\mathcal{I}}$, situation mining is divided into two phases, notably $\mathcal{SS}^{\mathcal{T}}$ *mining* and $SE^{\mathcal{T}}$ *mining*: Beginning with $\mathcal{SS}^{\mathcal{T}}$ *mining*, we first need to split each instance into its different situation states $s_t^{\mathcal{I}}$ (i.e., distinguish the individual *updates* of this situation). Thus, we temporally order all its encompassed object states by their distinct begin times. We define a situation state as the set of co-occurring object states, i.e., all object states active for the same time interval. Thus, we can obtain the different situation states $s_t^{\mathcal{I}}$ by splitting according to these time intervals. For instance, assume our situation S^1 comprises 5 different object states, denoted as o_1^1, o_2^1, o_3^1, o_2^2, o_3^2, following the notation $o_t^{objectID}$ with 3 different begin times $t=1$, $t=2$, $t=3$. Thus, we can discretize this situation instance into 3 different states:

$$\{o_1^1\}_{t=[1,2[} \quad \rightarrow \quad \{o_2^1, o_2^2\}_{t=[2,3[} \quad \rightarrow \quad \{o_3^1, o_3^2\}_{t=3}$$

Next, we need to derive the *object-relational signature* of each of these situation states, i.e., determine $\mathcal{SS}^{\mathcal{T}} := (\Omega, \rho)$. For each situation state $s_t^{\mathcal{I}}$, we analyze its encompassed object states o_t, to derive their object references $\mathcal{O}^{\mathcal{R}}$ (to form Ω), and the relation types $\mathcal{R}^{\mathcal{T}}$ holding between these o_t (to form ρ). For each unique identified object-relational signature, we create a corresponding $\mathcal{SS}^{\mathcal{T}}$, defined by these $\mathcal{O}^{\mathcal{R}}s$ and the $\mathcal{R}^{\mathcal{T}}s$ holding between them. For instance, for the above example, we might derive the following object types: o_1^1 and o_2^1 are of type ACI (accident), o_3^1 is of type ACX ("accident cleared"), and o_2^2 and o_3^2 are of type LS1 ("stationary traffic"), which results in the following $\mathcal{O}^{\mathcal{T}}$ sequence characterizing the evolution of this situation instance:

$$\begin{array}{ccccc} \underbrace{\{o_1^1\}_{t=[1,2[}} & \rightarrow & \underbrace{\{o_2^1, o_2^2\}_{t=[2,3[}} & \rightarrow & \underbrace{\{o_3^1, o_3^2\}_{t=3}} \\ \{ACI\} & \rightarrow & \{ACI, LS1\} & \rightarrow & \{ACX, LS1\} \end{array}$$

Furthermore, our $\mathcal{R}^{\mathcal{T}}$ analysis for the second and third state gives that o_2^1 and o_2^2 are spatially located on the same road segment, the same holds for o_3^1 and o_3^2. For this analysis, we simply test all $\mathcal{R}^{\mathcal{T}}s$ of the relation families of interest for the corresponding domain. In our road traffic scenario, we are testing the Simple Features topological $\mathcal{R}^{\mathcal{T}}s$ defined by the Open Geo-Spatial Consortium (OGC)[9], constrained to the underlying road network. $\mathcal{R}^{\mathcal{T}}s$ that evaluate to true are included in the $\mathcal{SS}^{\mathcal{T}}$ signature.

Thus, we can derive the $\mathcal{SS}^{\mathcal{T}}$ signature that for $\mathcal{SS}^{\mathcal{T}}$ {ACI,LS1}, two object states of type ACI and LS1 need to be co-occurring on the same road segment, to create a situation state instance of that type, analogously we can derive an $\mathcal{SS}^{\mathcal{T}}$ definition for {ACX,LS1}.

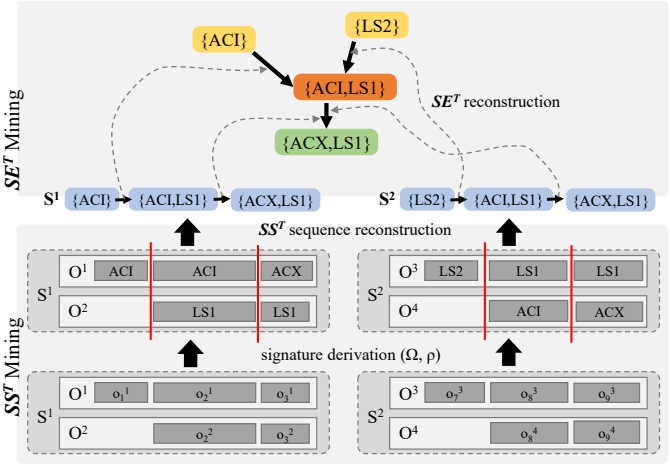


Figure 1: Illustrative situation mining example.

Algorithm 1 outlines this $\mathcal{SS}^{\mathcal{T}}$ *mining* step, which mainly bases on set, sequence, list and string comparisons. Since situations are analyzed individually, and only the resulting $\mathcal{SS}^{\mathcal{T}}$ signatures compared, $\mathcal{SS}^{\mathcal{T}}$ Mining is inherently suited for parallelization. The function ALIAS determines the $\mathcal{O}^{\mathcal{T}}$ of

[9]"geo:sf-disjoint", "geo:sf-touches", "geo:sf-overlaps", "geo:sf-equals", "geo:sf-within", "geo:sf-contains", "geo:sf-within", "geo:sf-intersects", "geo:sf-intersects", https://portal.opengeospatial.org/files/?artifact_id=44722

Algorithm 1 $\mathcal{SS}^{\mathcal{T}}$ Mining

Require:
S: A set of situation instances $\mathcal{S}^{\mathcal{I}}$
RTs: A set of relation types $\mathcal{R}^{\mathcal{T}}$ to check
Ensure:
$SSTs$: A set of $\mathcal{SS}^{\mathcal{T}}$s
$EvoSeqs$: A set of annotated situation instances $\mathcal{S}^{\mathcal{I}}$

1: **function** $\mathcal{SS}^{\mathcal{T}}$ INDUCTION(S, RTs)
2: $SSTs \leftarrow$ new Map<String, SST>( ) ▷ $\mathcal{SS}^{\mathcal{T}}$ index
3: $EvoSeqs \leftarrow \emptyset$ ▷ init. evolution sequences
4: **for all** $s \in S$ **do** ▷ loop over situation instances
5: $sitSeq \leftarrow <>$ ▷ init. current evolution sequence
6: $o \leftarrow s.o$ ▷ get all object states of this situation
7: $T \leftarrow$ ORDER(o.beginTime) ▷ extract times
8: **for all** t **in** T **do**: ▷ loop over begin times
9: $\widetilde{sst} \leftarrow$ new SST( ) ▷ new SST candidate
10: $o_t \leftarrow \{o \mid o.begintime = t\}$
11: $\widetilde{sst}.\Omega \leftarrow$ MAP($o_t \mapsto$ ALIAS(o_t)) ▷ get $\mathcal{O}^{\mathcal{R}}$s
12: $\widetilde{sst}.name \leftarrow$ MAP($o_t \mapsto o_t.OT$).toString( )
13: $\widetilde{sst}.\rho \leftarrow \emptyset$ ▷ initialize holding relations
14: **for all** $o_t^i \in o_t$ **do**
15: $a^i \leftarrow$ ALIAS(o_t^i) ▷ define alias
16: **for all** $o_t^j \in o_t, j \neq i$ **do**
17: $a^j \leftarrow$ ALIAS(o_t^j)
18: **for all** $r \in RTs$ **do** ▷ extract relations
19: **if** CHECK($r(o_t^i, o_t^j)$) = TRUE **then**
20: $\widetilde{sst}.\rho$.add($r(a^i, a^j)$)
21: $sign \leftarrow$ SIGNATURE($\widetilde{sst}$)
22: **if** $sign \in SSTs.keys$ **then**
23: $sst \leftarrow SSTs.get(sign)$ ▷ get existing
24: **else**
25: $SSTs.put(sign, \widetilde{sst})$ ▷ new SST found
26: $sst \leftarrow \widetilde{sst}$
27: sitSeq.append(sst) ▷ extend sequence
28: $EvoSeqs$.add(sitSeq)
29: **return** $SSTs.values$, $EvoSeqs$

an object state, and assigns it a unique name within this $s_t^{\mathcal{I}}$ (we basically create a "variable" for this object state $o_t^{\mathcal{I}}$)[10]. SIGNATURE converts an $\mathcal{SS}^{\mathcal{T}}$ to a unique textual representation of its composition (involving its $\mathcal{O}^{\mathcal{R}}$s and the $\mathcal{R}^{\mathcal{T}}$s holding between them).[11] For checking holding $\mathcal{R}^{\mathcal{T}}$s, Algorithm 1 assesses $\mathcal{R}^{\mathcal{T}}$s between object pairs, i.e., checks for binary relations. We note that unary $\mathcal{R}^{\mathcal{T}}$s may be incorporated as well, which only take a single object as input, and thus can be conceived as filters on a specific attribute of this object. Since any n-ary relation can be expressed by n binary relations [16], this construct can be generalized to arbitrary n-ary $\mathcal{R}^{\mathcal{T}}$s. Regarding the polynomial complexity of the $\mathcal{R}^{\mathcal{T}}$ computations (lines 14 to 20)[12], we note that real-world situations generally do not constitute a large number of involved objects and relations. Most $\mathcal{R}^{\mathcal{T}}$ families provide inherent potential for optimizing these comparisons: In particular, *symmetric* $\mathcal{R}^{\mathcal{T}}$s (i.e., $\rho(o_{ti}, o_{tj}) = \rho(o_{tj}, o_{ti})$), such as the topological $\mathcal{R}^{\mathcal{T}}$s we have employed in our case study, allow to halve the number of object comparisons (to $\frac{n^2-n}{2}$)[13]. Furthermore, $\mathcal{R}^{\mathcal{T}}$s such as spatial relations also might exhibit subsumption and mutual exclusiveness (e.g., spatially equal, "geo:sf-equals", subsumes overlapping, "geo:sf-overlaps", and automatically excludes disjoint, "geo:sf-disjoint"). Hence, by organizing the $\mathcal{R}^{\mathcal{T}}$ checking function in a principled way (in our implementation, lines 18 to 20 have been realized by a single, suitably hierarchically organized $\mathcal{R}^{\mathcal{T}}$ checking function), $\mathcal{R}^{\mathcal{T}}$ checking can be optimized.

As a result of $\mathcal{SS}^{\mathcal{T}}$ *mining*, we have created abstract descriptions for all unique observed co-occurring $\mathcal{O}^{\mathcal{T}}$ and $\mathcal{R}^{\mathcal{T}}$ combinations, i.e., identified the state space as the set of $\mathcal{SS}^{\mathcal{T}}$s, and represented each situation instance by the sequence of its traversed $\mathcal{SS}^{\mathcal{T}}$s.

In the subsequent $SE^{\mathcal{T}}$ *mining* step, described in Algorithm 2, we analyze these $\mathcal{SS}^{\mathcal{T}}$ sequences, to determine the observed transitions between $\mathcal{SS}^{\mathcal{T}}$s, which thus creates our $SE^{\mathcal{T}}$s by introducing a transition between two $\mathcal{SS}^{\mathcal{T}}$s for each observed $\mathcal{SS}^{\mathcal{T}}$ transition in these sequences (lines 4-9), i.e., identifying $\longrightarrow$ from Eq. 4. $SE^{\mathcal{T}}$s are created by processing all - not yet processed - $\mathcal{SS}^{\mathcal{T}}$s, and recursively following their incoming and outgoing transitions, until no further $\mathcal{SS}^{\mathcal{T}}$s can be reached (in graph theoretic terms, if we regard our state space as a graph, $SE^{\mathcal{T}}$ reconstruction corresponds to the identification of *connected components*, lines 10-23, whereby each connected component corresponds to a $SE^{\mathcal{T}}$). Thus, a $SE^{\mathcal{T}}$ is defined by the observed connectivity between $\mathcal{SS}^{\mathcal{T}}$s (i.e., the actually observed state evolutions). A difficult problem is how to find an explicative name characterizing the $SE^{\mathcal{T}}$'s content for a human – in the present approach, as a rough heuristic, the $SE^{\mathcal{T}}$'s name is the most frequently occurring $\mathcal{O}^{\mathcal{T}}$ (with randomly broken ties), which should indicate the main type of event prevalent in this $SE^{\mathcal{T}}$, or, for $SE^{\mathcal{T}}$s only comprised of a single $\mathcal{SS}^{\mathcal{T}}$, the name of the $\mathcal{SS}^{\mathcal{T}}$.

Fig. 1 illustrates the overall idea behind this fusion process.

V. Case Study Discussion

In the following, we discuss the results of applying our situation mining approach to the road traffic incident data described in Sec. II. In this proof-of-concept evaluation, we examine whether situation mining is capable of extracting domain-specific situation knowledge and generating meaningful $SE^{\mathcal{T}}$s. As can already be seen from the running example from Sec. IV, situation mining allows us to determine which events are semantically related, and how real-world situations actually evolve, i.e., which $\mathcal{O}^{\mathcal{T}}$ constellations typically follow each other. From this example (which is taken from our real-world data), we can automatically derive that accidents might

[10] This name is built from the object state's $\mathcal{O}^{\mathcal{T}}$, followed by a counter value in case the o_t contains multiple object states of this $\mathcal{O}^{\mathcal{T}}$. In the following, we may not specifically distinguish between $\mathcal{O}^{\mathcal{T}}$ and $\mathcal{O}^{\mathcal{R}}$, which correspond to the same string representation in $\mathcal{SS}^{\mathcal{T}}$s only comprised of distinct $\mathcal{O}^{\mathcal{T}}$s.

[11] Since Ω and ρ are sets and thus order-invariant, note that we impose an artificial ordering (lexicographic order) to obtain a consistent textual mapping.

[12] $\mathcal{O}((n^2 - n)\, r)$ for a situation state comprising n o_ts and testing r $\mathcal{R}^{\mathcal{T}}$s

[13] by indexing the object states and only comparing o_{ti} and o_{tj} if $j > i$ (i.e., if we express this cross-product $o_t \times o_t$ as a matrix indexed by i and j, this corresponds to checking the upper right triangular portion above the diagonal)

Algorithm 2 $SE^{\mathcal{T}}$ Mining

Require:
$SSTs$: A set of $\mathcal{SS}^{\mathcal{T}}s$
$EvoSeqs$: A set of $\mathcal{SS}^{\mathcal{T}}$ sequences (= annotated $\mathcal{S}^{\mathcal{I}}$)
Ensure:
$SETs$: A set of $SE^{\mathcal{T}}s$

function $SE^{\mathcal{T}}$ INDUCTION($SSTs, EvoSeqs$)
  for all $sst \in SSTs$ **do** ▷ init. transitions
    $sst.to \leftarrow \emptyset$, $sst.from \leftarrow \emptyset$
  for all sstSeq ∈ EvoSeqs **do** ▷ loop over instances
    for p=2, p++, p ≤ LENGTH(sstSeq) **do**
      $SST_p \leftarrow sstSeq[p]$ ▷ current position
      $SST_{p-1} \leftarrow sstSeq[p-1]$ ▷ previous pos.
      $SST_{p-1}.to.add(SST_p)$ ▷ add transitions
      $SST_p.from.add(SST_{p-1})$
  $queue \leftarrow$ ENQUEUE($SSTs$) ▷ build up SETs:
  for all $sst \in SSTs$ **do** ▷ loop over SSTs
    if $sst \in queue$ **then** ▷ not yet processed?
      $set \leftarrow$ new SET( )
      $set.ssts \leftarrow \emptyset$
      $nodesToExpand \leftarrow \{sst\}$
      ▷ recursively follow transitions to expand set:
      while $nodesToExpand \neq \emptyset$ **do**
        $c \leftarrow nodesToExpand.pop()$
        $set.ssts.add(c)$
        $expansion \leftarrow \{c.from, c.to\}$ ▷ expand
        for all $n \in expansion$ **do**
          if $n \notin set.ssts$ **then** ▷ yet processed?
            $nodesToExpand.push(n)$
      $SETs$.add(set) ▷ new SET constructed
      queue.remove(sst) ▷ already processed
  return $SETs$

trigger the build-up of traffic jams, which still persist even when the accident site has been cleared, thereby obviating the need to manually specify situation models encoding these event patterns for automated situation detection.

The large variety of $\mathcal{O}^{\mathcal{T}}s$ encountered in our road traffic application domain illustrates the benefits of such a knowledge mining approach. From our 1887 labeled situation instances, situation mining delivered 280 $\mathcal{SS}^{\mathcal{T}}s$, and 430 unique evolution sequences, i.e., possible paths through the resulting $SE^{\mathcal{T}}s$. Retrieved $\mathcal{SS}^{\mathcal{T}}s$ have a cardinality between 1 and 4, detailed results are provided in Table III. We note that $\mathcal{SS}^{\mathcal{T}}s$ of cardinality 1 do not perform any information fusion, since these only comprise one $\mathcal{O}^{\mathcal{T}}$. However, these have been included to obtain a unified representation structure for situations and enable a unified situation tracking across a situation's entire life-time (since situation states s_t of cardinality 1-$\mathcal{SS}^{\mathcal{T}}s$ may evolve to more complex situation states $s_{t\pm}$ of other $\mathcal{SS}^{\mathcal{T}}s$ at future time steps).

From our results, we observe that real-world situation definitions are actually *sparse* with respect to our given "alphabet" or input dimension, i.e., the set of $\mathcal{O}^{\mathcal{T}}s$. As we have 110 distinct $\mathcal{O}^{\mathcal{T}}s$, theoretically, we could generate 6105 $\mathcal{SS}^{\mathcal{T}}$ candidates of cardinality two (i.e., the cross-product $\mathcal{O}^{\mathcal{T}} \times \mathcal{O}^{\mathcal{T}}$, excluding symmetric cases), corresponding to the possible associations of two $\mathcal{O}^{\mathcal{T}}s$. However, the fact that in our real world data, we only observed 157 $\mathcal{SS}^{\mathcal{T}}s$ of cardinality two (corresponding to 2.6% of the theoretically possible combinations), indicates that for real-world situation instances, only very specific combinations of $\mathcal{O}^{\mathcal{T}}s$ are semantically meaningful (e.g., we might not require an $\mathcal{SS}^{\mathcal{T}}$ relating "vehicle on fire" with "flash floods", whereas we did extract the $\mathcal{SS}^{\mathcal{T}}$ {VFR,SMO}, expressing the meaningful event constellation "vehicle on fire" and "smoke hazard"), which we thus can uncover with situation mining. Thus, whereas potential $\mathcal{SS}^{\mathcal{T}}s$ could also be automatically generated as a cross-product of the set of $\mathcal{O}^{\mathcal{T}}s$, we observe that the largest part of this state space would correspond to combinations that are not semantically meaningful. Hence, this highlights the benefit of our empirical situation mining approach.

Analogously, we can examine the retrieved evolution patterns. Theoretically, 78400 transitions between $\mathcal{SS}^{\mathcal{T}}s$ would be possible in our state space, whereas we actually only observe 261 transitions (i.e., only 0.3% non-zero entries in the transition or adjacency matrix $\mathcal{SS}^{\mathcal{T}}s \times \mathcal{SS}^{\mathcal{T}}s$). As to be expected from the previous finding and aligning with our intuitive expectations, real-world situations apparently only expose a small set of relevant evolution patterns, which can be identified via situation mining. Regarding the observed connectivity, i.e., actually observed state evolutions, our resulting $SE^{\mathcal{T}}s$ appear overly fragmented, as the majority of $SE^{\mathcal{T}}s$ only comprises a single (evolutionary) state, whereas the few $SE^{\mathcal{T}}s$ comprising multiple $\mathcal{SS}^{\mathcal{T}}s$ jointly cover 50% of the $\mathcal{SS}^{\mathcal{T}}$ space. However, this may be owed to the actually rather low fraction of situation instances that indeed exposed evolving behavior (only 22% of $\mathcal{S}^{\mathcal{I}}$ comprised more than one s). Either, (i) these $\mathcal{S}^{\mathcal{I}}$ indeed did not show any object-relational development over time, or (ii) we might have missed some updates due the sampling frequency of our data recording[14], or (iii) this might be due to to incompletely recorded $\mathcal{S}^{\mathcal{I}}$, as operators might not have recorded all its distinct phases – this may be corroborated by the finding that in our $\mathcal{S}^{\mathcal{I}}$, we only rarely observe dedicated clearance messages that denote a situation's explicit resolution, such as ACX, ALL, CAL, TCX, VWX etc., whereas $\mathcal{S}^{\mathcal{I}}$ often abruptly end in any state. (i) and (ii) correspond to inherent characteristics of the data and the data set compilation, respectively, which thus need not be addressed in real-world application settings (in which we would have access to the complete set of situation records, without any sampling biases). Conversely, we note that case (iii) would motivate the need for supporting operators with automated situation detection and tracking means, to obtain more complete and consistent situation records, which facilitates situation analytics and situation prediction based on previous situation experiences.

VI. APPLICATIONS

A. Situation Analytics

As a direct outcome of situation mining, visualizing the resulting $SE^{\mathcal{T}}s$ allows us to perform *situation analytics*, i.e.,

[14]As described in Section II, we only received snapshots of currently observed states in 3-min. intervals.

	Counts	Fraction
$SE^{\mathcal{T}}$s	159	100%
# single-$\mathcal{SS}^{\mathcal{T}}$	140	88%
# multi-$\mathcal{SS}^{\mathcal{T}}$	19	12%
# $\mathcal{SS}^{\mathcal{T}}s$	280	100%
# cardinality 1	59	21%
# cardinality 2	157	56%
# cardinality 3	62	22%
# cardinality 4	2	0.7%
# Unique Evolution Sequences	430	
# Evolving Situations	407	22%

Table III: Situation Mining Results

we can perform *Investigative Situation Management* [17] by exploring the evolution patterns in our situation memory. Fig. 2 shows an excerpt of the largest $SE^{\mathcal{T}}$ retrieved in our case study, `LS1`, rendered with GraphViz[15]. As we observed in our resulting visualizations, situation labeling seems to show considerable variance: Apparently, different event codes have been used for denoting similar clearances of a situation. For instance, we encountered that both {VWX} ("vehicle on wrong carriageway has left") and {CAL} ("notification cleared") are successor-$\mathcal{SS}^{\mathcal{T}}s$ for the $\mathcal{SS}^{\mathcal{T}}$ {VWC} ("vehicle on wrong carriageway"), corresponding to the operator action of displaying messages on the VMS to notify approaching motorists that the previous wrong-way driver warning has been cleared, which denotes the resolution of a dangerous wrong-way driver situation. This corroborates findings reported in related work: Arco et al. have studied road traffic incident data from Italy, and also noted that apparently, different control center operators used different event codes for referring to semantically equivalent state of affairs [13]. However, this "operator-introduced" variance in the situation instances' resulting $\mathcal{SS}^{\mathcal{T}}$ sequences results in unnecessarily large $SE^{\mathcal{T}}s$, in which many $\mathcal{SS}^{\mathcal{T}}s$ are redundant in the sense that slightly different $\mathcal{O}^{\mathcal{T}}s$ are used for expressing a specific state of affairs. Hence, our situation mining approach allows to investigate - and consequently harmonize - such variance presumably introduced by the different preferences of different human operators for denoting such cases.

B. Situation Assessment

To relieve human operators from situation recording, we can employ our resulting $SE^{\mathcal{T}}s$ to directly generate the rule code for automatically detecting and tracking conforming situation instances further on. As illustrated in the previous section, it makes sense to precede this step by in-depth *situation analytics*, in which human domain experts can explore these models, to get an impression of how situation instances are currently composed, simplify these models (e.g., by fusing semantically equivalent $\mathcal{SS}^{\mathcal{T}}s$), which will produce situation instances exhibiting less "sequence variance", which would benefit situation prediction. Once domain experts have reviewed and revised the $SE^{\mathcal{T}}s$, model-to-text (M2T) transformations familiar from MDE can be employed, for automatically compiling each $\mathcal{SS}^{\mathcal{T}}$ to a corresponding situation detection rule. This can be provided for various rule engine implementations, as for each employed technology, only a corresponding M2T transformation for our $SE\mathcal{M}$ needs to be implemented. In our case study, which has been implemented in Java, we have developed a proof-of-concept M2T transformation for the JBoss Drools rule language[16]. Whereas we will omit the details of this M2T implementation, which is beyond the scope of the present work[17], we will report the main findings of our proof-of-concept evaluation. Our 280 $\mathcal{SS}^{\mathcal{T}}s$ could be automatically translated into corresponding Drools rules, such as the following Drools rule (simplified), which encodes the last $\mathcal{SS}^{\mathcal{T}}$ from our running example from Section IV:

Listing 1: Drools Rule Example for $\mathcal{SS}^{\mathcal{T}}$ {ACX,LS1}

```
rule "{ACX,LS1}" salience 2
    when
        $acx: AsfinagTrafficmessage ( datex_phr == "ACX" )
        $ls1: AsfinagTrafficmessage ( datex_phr == "LS1")
        eval($ls1.getBegintime().equals($acx.getBegintime()))
        eval($ls1.getRoad_code().equals($acx.getRoad_code()))
        eval($ls1.getBeginmeter() == $acx.getBeginmeter())
        eval($ls1.getEndmeter() == $acx.getEndmeter())
    then
        StateInstance ins = new StateInstance();
        ins.setName("{ACX,LS1}");
        ins.setBeginTime(currTime);
        ins.getAsfinagTrafficmessage().add($acx);
        retract($acx)
        ins.getAsfinagTrafficmessage().add($ls1);
        retract($ls1)
        insert(ins);
end
```

Based on our findings reported in Sections V and VI-A, we have implemented a different tracking approach than in previous work [11], as *situation analytics* has revealed that the identified $SE^{\mathcal{T}}s$ are unlikely to be yet complete due to the low fraction of evolving situations observed so far: Therefore, we employ an adaptive tracking approach based on a specific *situation update rule*, which fuses two temporally adjacent situation states $s^{\mathcal{I}}_{t-}$ and $s^{?}_{t}$ to $\mathcal{S}^{\mathcal{I}}$ whenever both share at least one $O^{\mathcal{J}}$, i.e., it compares their object states o_{t-} and o_t and finds two states $o^{\mathcal{J}}_{t-}$ and $o^{\mathcal{K}}_{t}$ such that $\mathcal{J} = \mathcal{K}$.[18] Hence, this tracking approach is capable of extending the $SE^{\mathcal{T}}s$ with additional transitions on-the-fly.

Running this SA implementation across the original data set allows to assess the correctness and completeness of the generated implementation. In our experiment, we have obtained 99% recall, i.e., our rules were generally capable of recovering the human-labeled situation instances[19], which is to be expected since our rules have been essentially "reverse-engineered" from these instances. For future work, we plan to examine the generalization capability of the mined models,

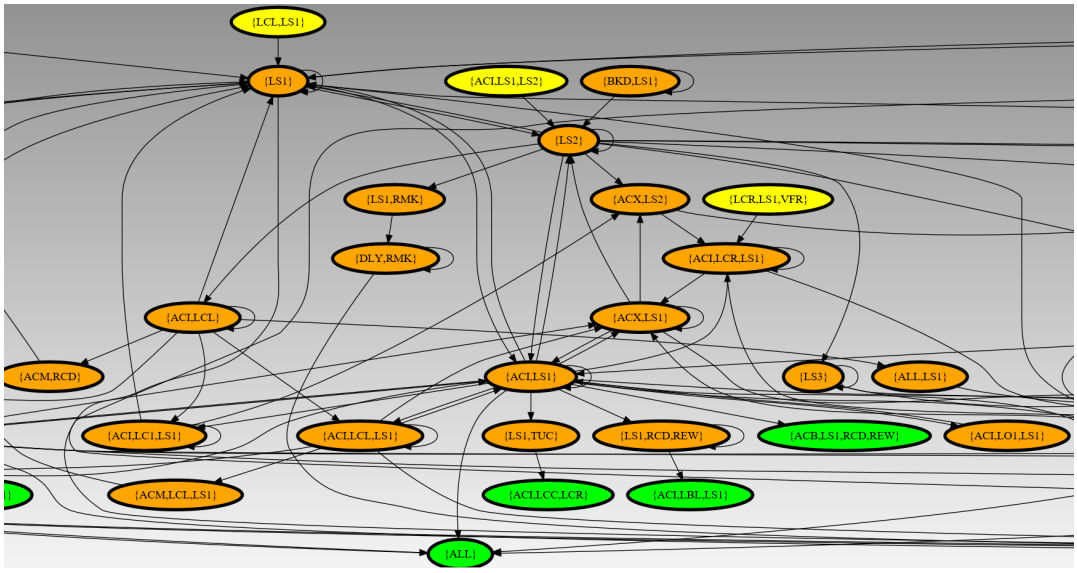
[15] `https://github.com/nidi3/graphviz-java`

[16] `https://www.drools.org`

[17] E.g., we need to consider the different cardinalities of the $\mathcal{SS}^{\mathcal{T}}s$ in this transformation, to account for subsumption relationships between $\mathcal{SS}^{\mathcal{T}}s$, as an event constellation matching $\mathcal{SS}^{\mathcal{T}}$ {ACI,LC1,LS1} would also match {ACI,LS1} and {LS1}, since {LS1} $\subset$ {ACI,LS1} $\subset$ {ACI,LC1,LS1}.

[18] This bases on the assumption that an object will simultaneously only participate in one situation, which holds for our road traffic incident scenario.

[19] For these tests, we could only use completely spatio-temporally specified situations, corresponding to 77% of our recorded $\mathcal{S}^{\mathcal{I}}$, as some or our recorded o had time or location slots unspecified, which would be required for automated SA. This was typically the case for events reported from the lower level street network that might impact highway traffic, which thus were only of general interest to ASFINAG, and thus often are not fully specified.



Legend [a]

Code	Designation
ACB	accident involving buses
ACI	accident
ACM	multi-vehicle accident
ACX*	accident cleared
ALL*	all accident sites cleared
BKD	broken-down vehicle
DLY	delays
LBL*	left lane blocked
LC1*	1 lane closed
LCC*	central lane closed
LCL*	left lane closed
LCR*	right lane closed
LO1*	only 1 lane open
LS1	stationary traffic
LS2	queuing traffic
LS3	slow traffic
RCD	road closed
REW	rescue & recovery work
RMK	maintenance work
TUC*	tunnel closed
VFR	vehicle on fire

Figure 2: Excerpt of the largest retrieved $SE^{\mathcal{T}}$, "LS1". We observe that the sequences of associated events intuitively make sense, such that accidents trigger the build-up of traffic jams and corresponding operator actions such as lane closures. *Situation mining* thus seems to be capable of acquiring such domain-specific situation knowledge in an automated fashion. Color coding: yellow = $\mathcal{SS}^{\mathcal{T}}s$ without incoming transitions, orange = $\mathcal{SS}^{\mathcal{T}}s$ with both incoming and outgoing transitions, green = $\mathcal{SS}^{\mathcal{T}}s$ without outgoing transitions.

[a]Mapping from DATEX II PIM v2.3 Data Dictionary (`https://www.datex2.eu/node/487`, sheet "Enumeration literals", columns "Original Code" & "Designation"). Codes marked with * do not appear in the Data Dictionary (thus might be ASFINAG-specific) and are mapped from their corresponding VMS messages.

by measuring recall on novel situation record data sets. Summarizing, our preliminary experiments highlight the potential of our situation mining approach, considering that otherwise 280 rules would have to be captured and specified manually to implement automated SA.

VII. Related Work

Despite the popularity of *deductive SA systems*, which require suitable logical specifications of the situations of interest to detect situation instances as event constellations matching these descriptions, only few approaches so far have aimed at supporting this *situation knowledge acquisition*. Human factors-based knowledge elicitation techniques have been adopted in [18], to collect evidence on real-world situations that should be modeled by means of participatory observations of maritime control center operators. Since the manual collection and analysis of operator actions is a time-consuming process, [19] proposed an experience logging tool to automatically record cyber analysts' interactions with the operational system. The logged event sequences and operator actions have then been mined to reconstruct the underlying mental situation models of the cyber analysts [20]. Hence, this approach for mining cyber situation models out of logged operator actions represents the most closely related work to our situation mining approach. However, the proposed mining approach is confined to the application domain of cyber-security analysis (e.g., by focusing on temporal $\mathcal{R}^{\mathcal{T}}s$ only, which are key for the studied network intrusion detection tasks, such as event A happened before event B), whereas situation mining generalizes to arbitrary object-relational domains, by allowing arbitrary $\mathcal{R}^{\mathcal{T}}$ families suitable for the domain at hand.

Inductive SA systems obviate human knowledge acquisition by compiling situation models from observation data. However, *sub-symbolic* statistical or machine learning models base on different meta-models, termed pattern or hypothesis languages (or model classes) [9] in machine learning terminology, than the human modeler. *Symbolic* machine learning approaches, which *induce logical event descriptions* from data, can be found in the area of non-monotonic Inductive Logic Programming (ILP). For instance, [21] derive symbolic event descriptions based on the Event Calculus by means of inductive-abductive reasoning. However, these approaches do not employ a specific meta-model like our $SE\mathcal{M}$ to constrain the structure of the resulting situation types and thus do not incorporate prior knowledge on the structure of the sought-after situation models. This is also the case for related data mining approaches, such as pattern-based spatio-temporal data mining approaches [22], which base on different pattern languages than our situation management-specific $SE\mathcal{M}$.

Probabilistic SA systems based on probabilistic graphical models (PGMs), such as (dynamic) Bayesian Networks [23], [24], Probabilistic Relational Models [25], and Hidden Markov Models [26], result in similar graph-based representations like our $SE^{\mathcal{T}}s$, however, express different semantics: The graph structures of PGMs represent the "global" conditional statistical dependence relations between random variables. Contrastingly, in our situation monitoring application, we are interested in describing and analyzing various $\mathcal{R}^{\mathcal{T}}s$ between our observed objects, thus identifying highly local patterns characterizing spatio-temporally correlated object constellations.

VIII. Conclusion and Future Work

In the present work, we proposed an approach for automated situation knowledge acquisition: We contributed a *situation mining* algorithm for reverse-engineering descriptive models

for different types of evolving situations, composed of a changing set of different types of objects, from already observed situation instances. We illustrated the benefits of generating such situation evolution types from "untyped" situation instances, which enables us to study the general evolution patterns in our situation memory, i.e., perform *situation analytics*, and automatically generate the corresponding situation detection and tracking code, so that corresponding instances can be detected automatically further on. Thus, our approach also demonstrates how descriptive data mining results can be automatically *deployed*, by compiling them to executable models for real-time detection and tracking.

Our current situation mining approach corresponds to a supervised data mining approach, since it requires labeled example situations to derive its models. For future work, we plan to extend this to an *unsupervised* situation mining approach, which does not rely on a-priori provided situation instances, but aims to identify potential situation instances autonomously by analyzing recurring patterns within spatio-temporal object-relational data sets, which may correspond to potential situation instances. In the presented feasibility study, we also generated a one-to-one mapping, i.e., each individual observed situation instance is compiled to corresponding $\mathcal{SS}^{\mathcal{T}}s$ (and subsequently detection rules) and transitions. However, this gives rise to large and fragmented $SE^{\mathcal{T}}s$, whereby many $\mathcal{SS}^{\mathcal{T}}s$ are redundant in the sense that slightly different event types are used to refer to a specific state of affairs. Hence, our approach allows to investigate - and further harmonize - such operator variance. To further support this harmonization, we plan to develop suitable $SE^{\mathcal{T}}$ pruning strategies by automatically detecting such semantic equivalences between individual $\mathcal{SS}^{\mathcal{T}}s$, and fusing them to a single $\mathcal{SS}^{\mathcal{T}}$. This consequently will reduce the variance within the $\mathcal{SS}^{\mathcal{T}}$ sequences of tracked situation instances, and thus improves situation prediction from previous experience, which we also plan to investigate.

ACKNOWLEDGMENTS

This work has been funded by the Austrian Science Fund (FWF) under grant FWF T961-N31. The analyzed data has been provided by ASFINAG.

REFERENCES

[1] C. Matheus, M. Kokar, K. Baclawski, J. Letkowski, C. Call, M. Hinman, J. Salerno, and D. Boulware, "Lessons learned from developing SAWA: a situation awareness assistant," in *8th International Conference on Information Fusion*, vol. 2, 2005.

[2] J. Edlund, M. Grönkvist, A. Lingvall, and E. Sviestins, "Rule-based situation assessment for sea surveillance," in *Proc. SPIE 6242, Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, vol. 6242, 2006, pp. 624 203–624 203–11.

[3] N. Baumgartner, S. Mitsch, A. Müller, W. Retschitzegger, A. Salfinger, and W. Schwinger, "A Tour of BeAware! – A situation awareness framework for control centers," *Information Fusion*, vol. 20, no. 0, pp. 155–173, 2014.

[4] V. K. Singh and R. Jain, *Situation Recognition Using EventShop*, 1st ed. Springer Publishing Company, Incorporated, 2016.

[5] M. Stocker, "Situation Awareness in Environmental Monitoring," Dissertation, University of Eastern Finland, Kuopio, Finland, Nov., 2015.

[6] G. D'Aniello, A. Gaeta, V. Loia, and F. Orciuoli, "Integrating GSO and SAW ontologies to enable Situation Awareness in Green Fleet Management," in *2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, 2016, pp. 138–144.

[7] M. Stocker, J. Nikander, H. Huitu, M. Jalli, M. Koistinen, M. Rönkkö, and M. Kolehmainen, "Representing Situational Knowledge for Disease Outbreaks in Agriculture," *Journal of Agricultural Informatics*, vol. 7, no. 2, pp. 29–39, 2016.

[8] I. Kadar, E. Bosse, J. Salerno, D. A. Lambert, S. Das, E. H. Ruspini, B. J. Rhodes, and J. Biermann, "Results from levels 2/3 fusion implementations: issues, challenges, retrospectives, and perspectives for the future an annotated perspective," pp. 696 812–696 812–34, 2008.

[9] J. Fürnkranz, D. Gamberger, and N. Lavrač, *Foundations of Rule Learning*, ser. Cognitive Technologies. Springer, 2012.

[10] M. Kokar, J. J. Letkowski, R. Dionne, and C. Matheus, "Situation tracking: The Concept and a Scenario," in *IEEE Military Communications Conference (MILCOM 2008)*, 2008, pp. 3175–3181.

[11] A. Salfinger, W. Retschitzegger, and W. Schwinger, "Staying Aware in an Evolving World — Specifying and Tracking Evolving Situations," in *Proceedings of the 2014 IEEE International Inter-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. IEEE, 2014, pp. 195–201.

[12] J. Llinas, C. Bowman, G. Rogova, A. Steinberg, E. Waltz, and F. White, "Revisiting the JDL Data Fusion Model II," in *Seventh International Conference on Information Fusion (FUSION 2004)*, 2004.

[13] E. Arco, A. Ajmar, F. Arneodo, and P. Boccardo, "An operational framework to integrate traffic message channel (TMC)in emergency mapping services (EMS)," *European Journal of Remote Sensing*, vol. 50, no. 1, pp. 478–495, 2017.

[14] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*, 1st ed. Morgan & Claypool Publishers, 2012.

[15] A. Salfinger, D. Neidhart, W. Retschitzegger, W. Schwinger, and S. Mitsch, "SEM^2 Suite -Towards a Tool Suite for Supporting Knowledge Management in Situation Awareness Systems," in *15th IEEE International Conference on Information Reuse and Integration (IRI 2014)*. IEEE, 2014, pp. 351–360.

[16] L. Cerf, J. Besson, C. Robardet, and J.-F. Boulicaut, "Closed patterns meet n -ary relations," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 1, pp. 1–36, 2009.

[17] G. Jakobson, J. Buford, and L. Lewis, "A Framework of Cognitive Situation Modeling and Recognition," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006.

[18] M. Nilsson, J. van Laere, T. Ziemke, and J. Edlund, "Extracting rules from expert operators to support situation awareness in maritime surveillance," in *2008 11th International Conference on Information Fusion*, 2008.

[19] C. Zhong, D. Samuel, J. Yen, P. Liu, R. Erbacher, S. Hutchinson, R. Etoty, H. Cam, and W. Glodek, "RankAOH: Context-driven Similarity-based Retrieval of Experiences in Cyber Analysis," in *2014 IEEE International Inter-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. San Antonio, TX, USA: IEEE, 2014, pp. 207–213.

[20] C. Zhong, J. Yen, P. Liu, and R. F. Erbacher, "Learning From Experts' Experience: Toward Automated Cyber Security Data Triage," *IEEE Systems Journal*, pp. 1–12, 2018.

[21] N. Katzouris, A. Artikis, and G. Paliouras, "Incremental learning of event definitions with Inductive Logic Programming," *Machine Learning*, vol. 100, no. 2, pp. 555–585, 2015.

[22] M. Celik, S. Shekhar, J. P. Rogers, and J. A. Shine, "Mixed-drove spatiotemporal co-occurrence pattern mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 10, pp. 1322–1335, 2008.

[23] Y. Fischer and J. Beyerer, "Defining dynamic Bayesian networks for probabilistic situation assessment," in *2012 15th International Conference on Information Fusion*, 2012, pp. 888–895.

[24] M. P. Jenkins and D. Young, "BARRACUDA: An augmented reality display for increased motorcyclist en route hazard awareness," in *2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, 2016.

[25] D. Meyer-Delius, C. Plagemann, G. Wichert, W. Feiten, G. Lawitzky, and W. Burgard, "A Probabilistic Relational Model for Characterizing Situations in Dynamic Multi-Agent Systems," in *Data Analysis, Machine Learning and Applications*. Springer Berlin Heidelberg, 2008.

[26] D. Meyer-Delius, C. Plagemann, and W. Burgard, "Probabilistic situation recognition for vehicular traffic scenarios," in *IEEE International Conference on Robotics and Automation, 2009. ICRA '09.*, 2009.