Towards a Scenario Database from Recorded Driving Data with Regular Expressions for Scenario Detection

Philip Elspas[1], Jonas Lindner[1], Mathis Brosowsky[1], Johannes Bach[1] and Eric Sax[2]
[1]*Dr. Ing. h.c. F. Porsche AG, Weissach, Germany*
[2]*Karlsruhe Institute of Technology, Karlsruhe, Germany*

Keywords: Scenario Database, Scenario Detection, Automated Driving, Data-driven Development.

Abstract: With increasing capabilities of Advanced Driver Assistance Systems (ADAS) and Automated Driving Systems (ADS) established automotive development processes are challenged. The specification phase faces an open world problem, with an exploding space of different driving situations and various corner cases. Scenario-based development provides a systematic approach to describe the operational design domain of ADAS and ADS with scenarios, that can be used along the development process until system qualification. However, deriving all relevant scenarios, that need to be considered remains an open challenge. Recorded driving data provides a valuable source of real-world scenarios with highest validity. A database with such scenarios can be used to validate requirements early in the specification phase. For system qualification, detected scenarios can be extended with test conditions or can be (re-)simulated. Furthermore, function development can leverage a scenario database for data-driven and machine learning methods. While a scenario database is a common concept most approaches remain abstract and vague in the description. In this work we analyze requirements and expectations on a scenarios database and propose a detailed design and concept. For the necessary scenario detection, we suggest a new method to identify complex pattern in multivariate time series based on regular expressions.

1 INTRODUCTION

Safety is a major concern in the automotive industry. As system failures can cause accidents or even fatalities, comprehensive testing should ensure a high level of safety. Therefore the Automotive Software Process Improvement and Capability Determination (Automotive SPICE) provides a process reference and assessment model for the development of automotive systems (VDA QMC Working Group 13 / Automotive SIG, 2017). The top level of the V-model requires a system requirement analysis on the left side, which is opposed by system qualification tests on the right side. Both, specified requirements and system qualification tests, must be linked in a traceable way. In other words: Each test case must be assigned to a requirement and each requirement must be verified with according tests.

With the open-world problem of highly automated driving the specification of system requirements becomes increasingly challenging due to the large number of possible driving situations. Scenario-based development addresses the issue by breaking down the overall driving task into scenarios. Then the proper driving behavior can be specified, developed and tested on a scenario level. The entirety of tested scenarios should cover the Operational Design Domain (ODD) and provide evidence for the systems safety.

In a survey on scenario-based safety assessment for automated vehicles Riedmaier et al. distinguish between a knowledege-based scenario generation and data-driven extraction of scenarios from driving data (Riedmaier et al., 2020). Different levels of abstraction and detail can be distinguished (Menzel et al., 2018): Functional scenarios provide a human understandable description, logical scenarios include parameter ranges and finally concrete scenarios are instances of logical scenarios. A major goal of knowledge-based approaches is the consistent usage of scenarios along those levels of abstraction. Menzel et al. suggest a key-word based approach to detail functional to logical scenarios (Menzel et al., 2019). Beyond the use of key-words, Bock et al. describe a domain specific language that supports consistent usage of functional, logical and concrete scenarios for

Elspas, P., Lindner, J., Brosowsky, M., Bach, J. and Sax, E.
Towards a Scenario Database from Recorded Driving Data with Regular Expressions for Scenario Detection.
DOI: 10.5220/0011085200003191
In *Proceedings of the 8th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2022)*, pages 400-409
ISBN: 978-989-758-573-9; ISSN: 2184-495X

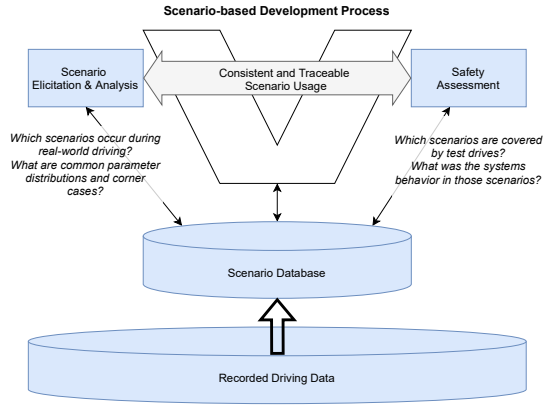

Figure 1: Leveraging recorded driving data within a scenario-based, automotive development process. On the left side the fundamental challenge of coverage needs to be addressed, on the right side large numbers of scenarios and test cases need to be evaluated.

systems engineering (Bock et al., 2019). Within the different development phases, recorded driving data can be leveraged: Bach et al. describe the use of recorded driving data for requirements elicitation and analysis, system and software design, unit construction and verification, and software and system integration and qualification (Bach et al., 2017).

Besides systems engineering challenges in a scenario-based development methodology, large amounts of recorded driving data lead to challenges in data management. Scenarios need to be labeled to identify relevant ones efficiently in hundreds or thousands of hours of driving data. Providing a way to store and retrieve information, that is convenient and efficient, is the primary goal of Database Management Systems (DBMSs) (Silberschatz et al., 2020). The management involves the definition of structures for storing information, providing mechanisms for manipulation and providing scalability to large amounts of information (Silberschatz et al., 2020). While the scenario database is a broadly used concept for scenario-based development (Bach et al., 2017; Pütz et al., 2017; Riedmaier et al., 2020), a concrete database structure for recorded driving data is hardly detailed. In this context, the main contribution of this work is twofold: First, we discuss a detailed database design and suggest a concrete data structure. Secondly, we propose a highly efficient, but flexible scenario detection as crucial enabler. A new method and algorithm for scenario detection supports complex pattern matching in multivariate time series. Such a scenario database can be used throughout the automotive development process, as indicated in Figure 1.

In Section 2 we start with the analysis of requirements and expectations on a scenario database. Based on these requirements we suggest a concrete structure for such a database in Section 3. Therefore scenario detection is a basic necessity and we use regular expressions as a formal model to define relevant data patterns. This concept was introduced in (Elspas et al., 2020), but is further detailed and extended in Section 4 to deal with patterns in multivariate time series. The concept is exemplarily evaluated in Section 5 and we compare our concept with with state of the art approaches in Section 6.

2 REQUIREMENTS ON A SCENARIO DATABASE FOR DRIVING DATA

Setting up a scenario database in practice is challenging. Recorded driving data includes information from perception systems, sensor fusion, planning and prediction modules and the controllers of driving functions. Bottom-up approaches, where repeating and similar patterns are clustered to receive interpretable groups of scenarios have to deal with the high dimensionality of recorded driving data. Current unsupervised approaches explicitly select meaningful signals before identifying clusters in the data. So Montanari et al. cluster groups of lateral driving maneuvers (Montanari et al., 2020), Langner et al. cluster similar road segments (Langner et al., 2019) and Ries et al. reduce the driving state to a few, binned signals, before similar driving sequences are identified via an adaption of word embeddings (Ries et al., 2019). Consequently the found clusters are biased towards the feature engineering and unsupervised scenario detection seems to be still limited to cluster relatively few features of a scenario.

Within a scenario database a broad number of scenario features should be obtainable. In contrast to unsupervised methods, rule-based approaches are deterministic, provide interpretability and can be used to identify scenario features in recorded driving data as well (Elspas et al., 2020; Montanari et al., 2021; de Gelder et al., 2020). With the proper data structures and methods, such scenario features can be efficiently combined to scenarios, as we will further elaborate in this work.

2.1 Recorded Driving Data

Current vehicles have a large number of distributed Electronic Control Units (ECUs), that communicate and exchange information via signals on different

Figure 2: Trace data with a stream of messages. No direct access to the signals is supported.

bus systems, like CAN or FlexRay. Data recording is commonly done by logging all messages on the bus systems. Especially odometry data, map data, detected lane markings, traffic signs and other traffic participants from the perception systems provide valuable scenario information. However, such traces are highly unstructured data, as indicated in Figure 2, and there is no direct access to individual signals. For large scale aggregations and data analytics direct access to the dedicated signals is efficient and a desirable data interface.

But also data enrichment is necessary. Restricted band width, low latency and limited compute power within the vehicle lead to groups of signals that are highly optimized and might require extensive domain knowledge for a proper interpretation. Therefore recorded driving data needs to be enriched by smoothed or filtered signals and by decoded signal groups with less efficient, but better understandable information. Furthermore the advantage of hindsight and larger compute capabilities allow powerful processing and pattern matching to identify abstracted scenario information.

2.2 Scenario Model

The major goal of a scenario database is the convenient, efficient and intuitive access to scenarios from recorded driving data. However, there is neither a complete set of possible scenarios, nor could those scenarios be objectively identified in recorded data. A limited field of view, sensor noise, perception uncertainty and even erroneous data make the scenario detection challenging. According to the commonly used definition from Ulbrich et al. a scenario is the temporal development of scenes and a scene is the snapshot of the environment including scenery and dynamic elements (Ulbrich et al., 2015). In the real world scenes are incomplete, uncertain or even incorrect. For the description and representation of scenes a 5 Layer model can be used to distinguish features belonging to the road level (Layer 1), traffic infrastructure (Layer 2), temporal modifications of the previous layers (Layer 3), Objects (Layer 4) and Environment (Layer 5) (PEGASUS Project, 2019; Bagschik et al., 2018). Future work added a sixth layer for digital information (Bock et al., 2018; Scholtes et al., 2021). Those layers provide a structure for the features that represent singular aspects of scenes and scenarios. In

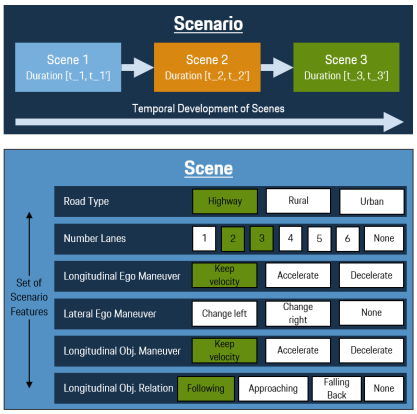

Figure 3: A scenario is the development of scenes (Ulbrich et al., 2015). A scene can be defined with a morphological matrix of scenario features.

the following we refer to those as “features” or “scenario features”.

De Gelder and Op den Camp suggest the use of tags, arranged in multiple tree structures, to describe these features (de Gelder and Camp, 2020). In a flattened structure, such features can be arranged in a morphological matrix, as exemplarily shown in Figure 3, for a simple interface to describe scenes.

An issue of such models is the quickly exploding scenario space it describes. The shown example can already represent 2.268 different scenes that lead to over 5 million different scenarios consisting of 2 scenes. The question for all relevant scenarios in the context of a combinatorially exploding scenario space is broadly raised and discussed in research (de Gelder et al., 2019; Hauer et al., 2019; Koopman and Fratrik, 2019; Hartjen et al., 2020).

Since there is no simple answer to a complete set of scenarios, an extendible and flexible database design is necessary. Scenario features, that provide the basic building blocks of a scenario, should not only be an intermediate processing step, but a substantial part of the database. Those could be used for data exploration on a feature level and support iterative and data-driven assembly of scenario catalogues.

2.3 Scalability and Versioning

Besides the expectations on a formal scenario model and the need of structured data, large amounts of recorded driving data require proper data management. Scalability is needed to handle increasing amounts of data as well as increasing sets of features and scenarios. Independent data tables, so called relations, for all the different features and scenarios can be used to support multiple parallel processing and scenario detection tool chains. With changing sce-

nario parametrization during the development, also versioning concepts are needed as Automotive SPICE requires traceability along the development phases.

2.4 Summary

As discussed in this section, a scenario database has to address challenges and expectations from different engineering field. In summary, we see 3 major points of view, that are addressed by our proposed concept:

1. Raw data traces alone are hardly sufficient for analyzing large amounts of data. Therefore the data needs to be structured and provide efficient interfaces for flexible and scalable data access.
2. Scenario specification and scenario-based testing requires a higher-level of abstraction than the raw signals. Enrichment with scenario features can provide a more intuitive and desirable interface.
3. Data Management has to deal with a large and growing number of scenarios and scenario features. Together with versioning requirements, a modular concept is needed to avoid complicated dependency structures.

3 A HYBRID DATABASE FOR TIME SERIES AND INTERVALS

Temporal data, such as recorded driving data can be represented in different forms. In contrast to data traces, multivariate time series provide values for each, discrete time step. An other representation are time intervals. Instead of providing the values for each time step, each value is saved along with a start time and an end time. In the given context, the value can be interpreted as a (scenario-) label. Time intervals can be easily extended with further attributes. Besides the duration, indicated by start and end time, also the driven distance or the mean velocity could be added to each interval. This provides a powerful representation for large scale aggregations: The time intervals can be grouped by the label and attributes can be aggregated or used for further filtering. Consequently both representations, time series and time intervals have their eligibility. For a scenario database, we suggest a hybrid database containing time series along with time intervals to leverage advantages from both representations.

3.1 Data Structure

Data traces are hardly efficient for large scale data analytics. For this purpose relevant signals can be indexed or extracted in a different data format. In this work we suggest the use of a columnar storage format, where a unique file id and the timestamp form the primary key, which is a unique identifier of each data row. The extraction of signals into a columnar storage of time series is shown in Figure 4 as the data preparation step. To align the signals, that are sent consecutively over the bus interface in the vehicle we round the timestamps to 10 ms and use a forward fill.

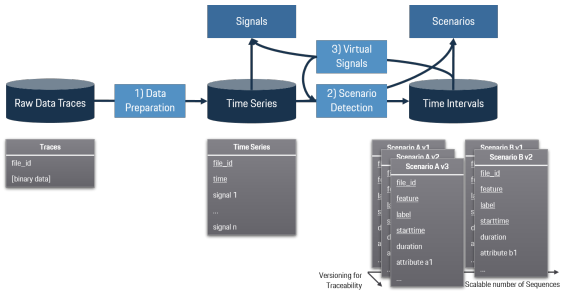

Figure 4: Concept for a Hybrid Scenario Database with Time Series and Time Intervals to represent scenarios. Time Intervals can be stored in independent relations to simplify scaling and versioning.

While a time series database can already provide various data statistics, a scenario-based development requires data access on an event level. This means a single scenario, covering a certain time span, should be represented as a single entry in the database. Scenario detection can derive time intervals as a convenient data representation as shown in Figure 4. Start and end times of a scenario can be saved and further attributes like the mean velocity or maximum acceleration can provide valuable scenario attributes to analyze parameter distributions or further filtering.

Finally, each detected scenario provides valuable information, that could be used to derive further or combined scenarios. To avoid a detection logic that has to deal with the input of time series and intervals we suggest the concept of virtual signals, as outlined in Figure 4. Time intervals can be dynamically read as time series and are therefore available to all applications using the signal representation. This includes the scenario detection so that detected scenarios can be simply referenced and reused in the detection logic. Note, that the extensive use of virtual signals can cause complex dependency structures and we suggest a limitation to two levels of abstraction as detailed in the following section.

3.2 Scenario Features and Scenarios

As described in Section 2.2, a scenario is a composition of multiple scenario features and their temporal development. Not all scenario features are di-

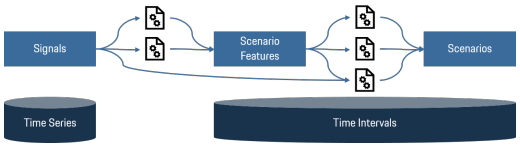

Figure 5: Concept to decouple the detection of scenario-features and scenarios. The scenario features are represented as time intervals but can be read as time series dynamically to support the same processing logic in both steps.

rectly available in the recorded driving data. Identifying meaningful and intuitive features can be done by simple mappings, decoding and combining groups of signal, smoothing and filtering. Rule-based approaches and pattern matching can be used to identify more complex features or driving maneuvers like cut-ins. The correct interpretation and processing of the available signals might need significant domain knowledge.

To reduce the complexity for a scenario engineer to deal not only with multiple dimensions of a scenario, but also the mapping of desired scenario features to the available information in recorded driving data, we suggest a 2-step process, as shown in Figure 5. First the recorded driving data is abstracted to a set of scenario features. This can be done by a number of independent feature detectors. All the processing steps are defined in configuration files that can be versioned and can be used to reproduce the detection results. The detected features are stored as time intervals. In contrast to appending the detected features as derived signals to the time series, the time interval representation supports scalable and convenient data access within the feature space. This way statistics about the frequency and duration or even the distribution of feature attributes can be evaluated even before the combination to scenarios.

Finally the scenario features provide an abstracted data representation and support the scenario definition. The available features can be arranged in a morphological matrix, as in Figure 3, or provide the basis for Graphical User Interfaces (GUIs) to define scenarios without further programming or data processing knowledge. Figure 6 shows an example GUI where a scenario is defined as the sequence of 5 distinct scenes based on 3 features for lateral, longitudinal maneuvers of the ego vehicle and cut-ins. For the practical realization, the concept of virtual signals is crucial: It allows dynamically reading the identified features, represented as time intervals, as time series. Then, the same framework and methods can be used to detect features and scenarios.

4 REGULAR EXPRESSIONS FOR PATTERN DETECTION IN MULTIVARIATE TIME SERIES

The proposed concept of a hybrid scenario database is strongly based on the capability to identify features and scenarios in time series data. Therefore we suggest a method that leverages regular expression for matching temporal patterns in multivariate time series. While the basic idea of using regular expression for scenario detection was already described in (Elspas et al., 2020), we further detail the concept to support multivariate patterns while keeping a simple interface. The method is in accordance with the scenario definition as temporal development of scenes (Section 2.2). For simplicity we describe the scenario detection for the combination of features to scenes and scenarios, even though the method can be used equally for the detection of features from the raw driving signals.

4.1 Basic Approach

As interface for describing scenarios we use an ordered list of scenes. Each scene is defined by a boolean expression based on one or multiple signals from the input data. Furthermore a minimum and maximum duration define the number of time steps that should be matched for a valid pattern.

Now the scenario detection can step through the recorded driving data and evaluate the scene conditions: If the current scene condition is satisfied, then the next time step is evaluated with the next scene condition in the scenario pattern. If the scene duration is longer than one time step the same condition might be evaluated multiple times. If all scenes of a scenario pattern were positively evaluated, the corresponding sequence can be marked as a matched scenario.

However, this simple algorithm gets more complicated when scenes do not have a fixed, but a variable length. With flexible scene lengths large numbers of concrete scenario pattern become valid and more dedicated search algorithms are needed. In this work, we suggest the use of regular expressions with recursive backtracking as a mature and formalized method for pattern detection in sequential data.

4.2 Regular Expressions

Regular expressions (regex), as described in detail in (Friedl, 2006), are widely used in text-processing and search interfaces as a powerful tool for pattern matching. They are rooted in theoretical computer science as a part of formal language theory where they

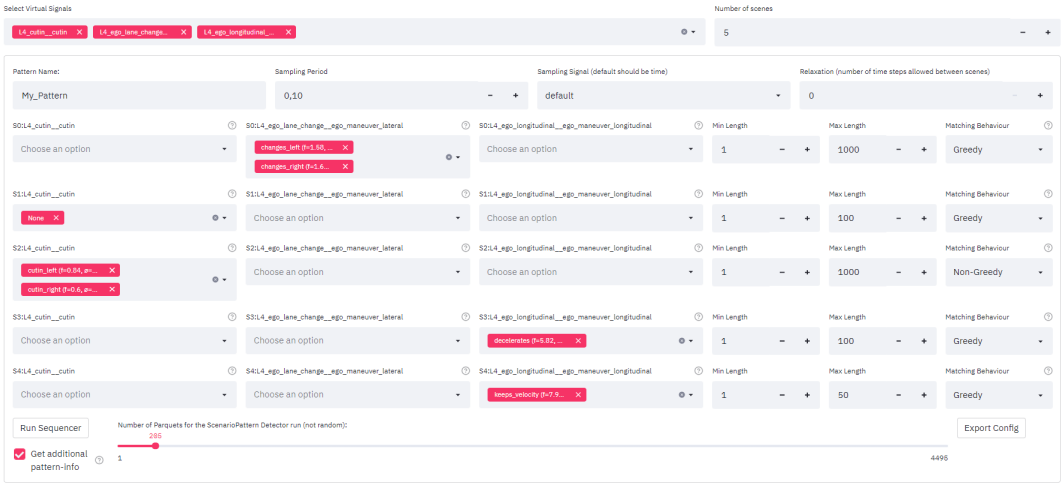

Figure 6: GUI for designing a scenario from pre-detected features. First a set of features and the number of scenes can be selected. Then each scene can be configured with concrete values for each feature and a flexible duration. The configured detector can be directly executed on recorded data or exported for deployment.

describe regular languages. Regex provide a compact interface to express sequential pattern. So called quantifiers can be used to describe repeating occurrences of a symbol. We use these to model scene durations with a minimal and a maximal number of time steps to provide high flexibility in the scenario definition.

When using scenes of flexible duration, it must be distinguished between greedy or lazy search patterns. Greedy means that the maximum scene length is tried to match before continuing with the next scene. If matching fails, the scene length is reduced step by step until a valid scenario is found or the minimum scene length is reached. A lazy, or non-greedy, search would evaluate the next scene directly after the minimum scene length was matched and only extend the scene length if the remaining scenario pattern was not matched. Both search methods can result in different detection results and should be considered.

4.3 Interface and Algorithm

While regular expressions provide the temporal concepts that are needed for scenario detection, they operate on strings. However, recorded driving data are multivariate time series of numeric, categorical, boolean or even mixed type. To apply regex, each time step in multivariate time series can be mapped to a state variable and encoded as a single character. By concatenating those characters, recorded driving data can be represented as string.

As a high level interface we use an ordered list of scenes to define a scenario. Within a python based framework for scenario detection a boolean condition, based on one or more signals, and minimum and maximum duration are defined in a configuration file. Greediness is supported via a boolean flag. The interface for an exemplary scenario with three scenes is shown in Figure 7 on the left. While the user of this interface does not have to deal with regex at all, internally the parameters are arranged within a second, regex based interface as shown in Figure 7 on the right. The scene conditions are mapped to an alphabet and the durations to regex operators. Also greediness is supported by the according regex operator. Advanced users could also directly define the regex pattern and leverage the full expressive power of regular expressions.

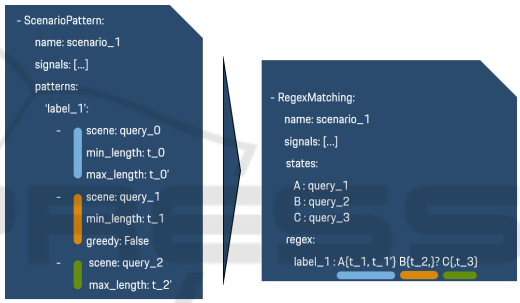

Figure 7: Scenario detectors can be configured as a list of scenes with minimum and maximum duration (left) and are automatically composed to a regular expression, that is matched within the data (right). Also further regex operators can be used within the right interface.

If all the defined scenes are mutually exclusive, each time step can be assigned to a unique state. However, the exclusiveness of scenes is a limiting requirement and we use a further mapping to support also non exclusive states: For each time step all state con-

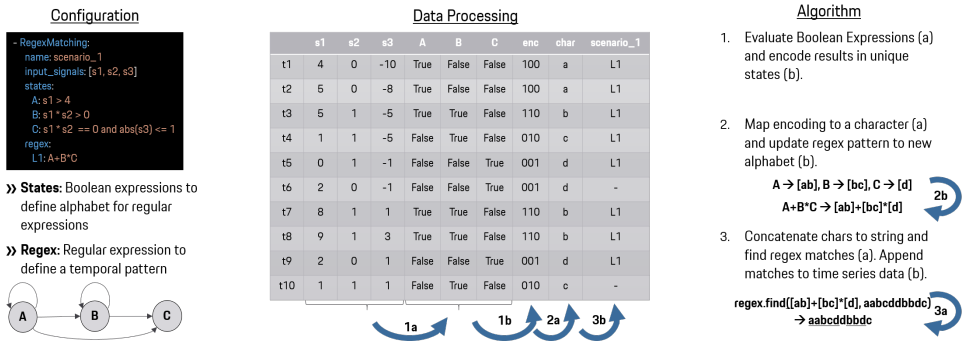

Figure 8: The algorithm for scenario detection with regular expressions: Based on a scenario configuration the input signals are processed in 3 major steps to receive labels for all matched scenarios. The encoding as a boolean state vector with the mapping to a new alphabet supports multivariate pattern.

ditions are evaluated and create a boolean state vector. Now each distinct state of the boolean vector can be dynamically assigned to a character. Before finding all matches of a given regex, also the regular expression itself needs to be adapted to the new alphabet.

Formally, the input is a set of conditions described by boolean expressions $q_1, \ldots, q_n$ with their labels $\Sigma = \{A, B, C, \ldots\}$ and a regular expression r over Σ. Now, the pattern detection consists of three steps, as exemplarily shown in Figure 8:

1. All boolean state conditions are evaluate and append the input data with new boolean columns.
2. Each time step is labeled with a new char encoding Σ' such that each combination of truths for the boolean expressions is uniquely encoded by a single character. The input regex r is updated to r' over Σ'. To do so each $A \in \Sigma$ is replaced with the set of characters from Σ' which encode a truth combination where A's boolean expression holds.
3. The time series encoding in Σ' is concatenated to a string and all occurrences of r' are matched. The start and end positions are mapped back to time steps.

This algorithm links an abstracted scenario definition with mature pattern matching. Flexible time ranges of scenes are supported. Even further regex concepts, like look ahead and look back operators, can be used to define contemporary scenes or extend scenario detectors with the evaluation of an expected system behavior. A further discussion of all the capabilities of regular expressions is out of the scope of this work and is left to future usage of the method.

5 INTERACTIVE AND DATA-DRIVEN SCENARIO ANALYTICS

Deriving knowledge from recorded driving data is an iterative process. General reference processes like the Knowledge Discovery in Databases (KDD) (Fayyad et al., 1996) or CRoss Industry Standard Process for Data Mining (CRISP-DM) (Wirth and Hipp, 2000) stress the loops, where data mining leads to increased knowledge and refinement of the initial problem in the next process iteration. This concept also applies for analyzing recorded driving data for a scenario-based development. In this section we show exemplary how the proposed scenario database and scenario detection enables fast data analytics that allow highly interactive workflows.

5.1 Performance Benchmark

To demonstrate the speed benefits of a data format that is not optimized for recording bus data in real time, but for fast read access, we consider a simple statistical aggregation of the detected lane marking type for different road types.

The aggregated driving duration for each road type and lane marking type is shown in Figure 9 and is based on 340 hours of fully logged driving data, which account for 5.3 Terabyte of recorded data. Neglecting network traffic to access such data from a scalable cloud storage, a simple compute node would need over 10 hours for just reading the necessary two signals. By structuring the data and extracting 1500 most relevant signals from the raw data requires only 18 GB storage. Also processing shows a significant speedup. The exemplary analysis evaluates 2 signals from 1.2 billion time steps and takes roughly 15 minutes on a single compute node, which is magnitudes

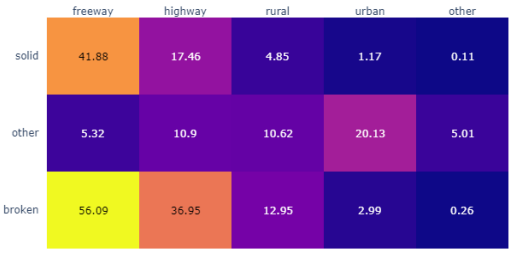

Figure 9: Heatmap with the aggregated duration in hours grouped by road type and lane marking type.

faster than just reading the raw data. Note, that this analysis if performed on the time series representation, by extracting the relevant signal into time intervals, the final aggregation can be performed within seconds.

5.2 Relaxation and Sub-scenarios

Working with the scenario database, and especially using the GUI shown in Figure 6, we noticed that many naively defined scenarios are hardly found in the recorded data. One reason for this is the quiet rigid scenario model that requires exact matches. In practice noisy data, temporarily missing information or the occurrence of short, intermediate scenes, that were not considered, can cause finding hardly any scenarios in the data. If this is not intended, short, optional gaps between all scenes of a scenario can be added. We call this concept *relaxation* and integrate it into the regex matching algorithm. With a relaxation parameter defining the maximum number of time steps, a regex wildcard symbol is added between all scenes and is matched non-greedily to find also similar scenarios with short intermediate scenes. For example the regex pattern A^+B^*C becomes $A^+.\{,x\}?B^*.\{,x\}?C$ for the relaxation value x. In regex syntax the "." symbol matches any state, while the "?" denotes non-greediness.

We demonstrate the power of this idea by considering a "follow" scenario where the ego-vehicle reacts to the acceleration by the leading vehicle:

- Scene 1: The Ego-vehicle has a constant distance to the leading vehicle and keeps the velocity.
- Scene 2: The distance to the leading vehicle grows and the ego-vehicle accelerates.

Searching this scenario in 10 hours of driving data takes 30 seconds and yields 2 scenarios. Increasing the relaxation parameter yields a significantly increasing number of scenarios as shown in the following table:

Relaxation (in s)	0	0.1	0.3	0.5	1	2
Number of matches	2	4	7	15	22	36

Such analysis can be performed relatively fast and provides a method to trade off ideal scenarios versus their occurrence rate during real world driving.

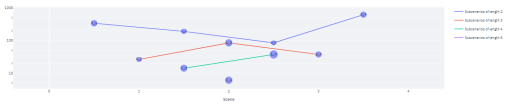

Figure 10: Number of occurrences for all sub-scenarios of a scenario with 5 scenes.

Also larger number of scenes can cause finding only few scenarios. Sub-scenarios can be searched and analyzed to identify the scene changes that cause a low number of detections. Figure 10 shows the number of occurrences of all sub-scenarios. This means the occurrences of all 2-scene, 3-scene and 4-scene scenarios within the 5-scene scenario are identified and counted. Such analysis can reveal the critical scene changes that might reduce the number of scenarios significantly. Also simple scene statistics about the minimum, maximum and mean duration of scenes can be quickly aggregated and provide useful insights.

While such statistics allow no general conclusion, evaluating detectors quickly and providing additional information and aggregations from the data is the key capability of a data-driven validation of scenarios within the requirements elicitation and analysis phase of the development process.

6 COMPARISON WITH RELATED WORK

A scenario database is not a new idea. Bach et al. use a scenario database for data-driven development (Bach et al., 2017) and Riedmaier et al. structure their survey on scenario-based safety assessment by methods that supply scenarios and methods that extract scenarios from a scenario database (Riedmaier et al., 2020). However, a concrete database design is out of their scope. For a database of relevant scenarios Pütz et al. suggest a 7-step data processing chain (Pütz et al., 2017). Several of these steps can be found in our concept as well. Data harmonization and transformation are included in the data preparation where time series are extracted from raw data traces. However, we leave most processing steps, like renaming signals or the generation of deduced signals to the later scenario detection. Furthermore we do the calculation of deduced signals dynamically, as this simplifies the data management and the usability for multiple use cases. A major difference can be seen in the scenario detection: Pütz et al. calculate scenario likelihoods over time and extract snippets with a high likelihoods. However, this approach hardly complies

with the scenario definition by Ulbrich et al. where a scenario is defined as the temporal development of scenes (Ulbrich et al., 2015). Defining a continuous scenario probability for a sequence of scenes is hardly intuitive. Our approach to define scenes with a boolean condition, based on (derived) signals from recorded data, and defining a scenario as a sequence of scenes incorporates the scenario definition by Ulbrich fundamentally. The use of regular expressions provides a concrete method for the implementation of the scenario detection.

The use of regular expressions for scenario detection was already introduced in (Elspas et al., 2020). However, the scenes were expected to be mutually exclusive so that the input data could be reduced to a time series. In this work we further detailed the concept and suggested a new algorithm that is able to deal with contemporary scenes. The support of multivariate patterns allows a more flexible and intuitive definition of scenario detectors.

A general discussion of knowledge mining in time series was done by Mörchen (Mörchen, 2006): In contrast to Allen's Interval Operators, that are commonly used to describe the relationship between intervals (Allen, 1983), Mörchen introduces a more robust time series knowledge representation (TSKR). In this formalism "Tones" provide basic time intervals, similar to scenario features in the context of driving data. Chords indicate the simultaneous occurrence of Tones, similar to scenes. Finally phrases express a partial order of chords and describe an abstraction similar to scenarios on a more general pattern level. While Mörchen suggests a method to find phrases that is based on the support of chords, we suggest the use of regular expressions to leverage domain knowledge about scenarios of interest within a well defined and flexible formalism.

Regex matching in multivariate time series has also be used in (Rodrigues et al., 2019). In contrast to our approach, the authors create the symbolic string for regex matching by appending the symbolic states from each input signal in an alternating way. We argue, that this adds cognitive load to the domain experts who has to define a regex pattern across different alphabets of the input series. Also the length of the string grows linear with each input signal, which can cause bad performance. Our approach maps the boolean scene conditions dynamically on a new alphabet and keeps the input length and the computational complexity. A disadvantage of our approach might be the limited alphabet. The number of distinct states in the target alphabet grows exponentially with the number of boolean scene conditions. Due to the dynamic approach to map only occurring states, we did not reach this theoretical limitation in practice.

7 CONCLUSION AND OUTLOOK

Scenario-based development is a crucial enabler for increasing capabilities of ADAS and ADS. For safety assessment and system qualification with highest validity, scenarios from real world driving data are a necessity. While the concept of a scenario database is often mentioned in literature, the necessary data structures remain vague or require already fully defined scenarios. However, we argue that recorded driving data should also be leveraged to derive meaningful scenario specifications in the first place. Therefore we proposed a highly flexible database design, that allows decoupling scenario features and supports fast and interactive queries. The advantages of time series and time intervals are combined by a hybrid database structure that allows dynamic transformation between both representations. Our performance evaluations demonstrates the efficiency for a broad spectrum of tasks from simple queries and aggregations of time series, over the recombination of scenario features to scenarios, up to identifying similar scenarios or sub-scenarios. With the proposed design many hours of driving data can be analyzed within few minutes, which enables an interactive and data-driven development.

While this work describes a concept for data management along with a method for domain experts to define and find scenarios of interest, more data-driven approaches can provide a valuable complement. So it was shown in (Elspas et al., 2021), that neural networks can be trained for scenario detection with weak labels from rule-based approaches to identify similar and additional scenarios. Saturation effects, as in (Hartjen et al., 2020), could be used to estimate the scenario coverage or various forms of anomaly detection could identify corner cases. Nevertheless, an efficient and performant database provides a valuable and beneficial basis for further research with recorded driving data.

REFERENCES

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11).

Bach, J., Langner, J., Otten, S., Holzäpfel, M., and Sax, E. (2017). Data-driven development, a complementing approach for automotive systems engineering.

Bagschik, G., Menzel, T., and Maurer, M. (2018). Ontology based scene creation for the development of auto-

mated vehicles. *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018-June:1813–1820.

Bock, F., Sippl, C., Heinz, A., Lauer, C., and German, R. (2019). Advantageous usage of textual domain-specific languages for scenario-driven development of automated driving functions. In *2019 IEEE International Systems Conference, SysCon 2019*. IEEE.

Bock, J., Krajewski, R., Eckstein, L., Klimke, J., Sauerbier, J., and Zlocki, A. (2018). Data basis for scenario-based validation of had on highways. In *27th Aachen colloquium automobile and engine technology*.

de Gelder, E. and Camp, O. O. d. (2020). Tagging real-world scenarios for the assessment of autonomous vehicles. *arXiv preprint arXiv:2012.01081*.

de Gelder, E., Manders, J., Grappiolo, C., Paardekooper, J.-P., Camp, O. O. d., and De Schutter, B. (2020). Real-world scenario mining for the assessment of automated vehicles. *arXiv preprint arXiv:2006.00483*.

de Gelder, E., Paardekooper, J.-P., Op den Camp, O., and De Schutter, B. (2019). Safety assessment of automated vehicles: how to determine whether we have collected enough field data? *Traffic injury prevention*, 20(sup1).

Elspas, P., Klose, Y., Isele, S. T., Bach, J., and Sax, E. (2021). Time series segmentation for driving scenario detection with fully convolutional networks. In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems, VEHITS 2021*. SCITEPRESS.

Elspas, P., Langner, J., Aydinbas, M., Bach, J., and Sax, E. (2020). Leveraging regular expressions for flexible scenario detection in recorded driving data. In *2020 IEEE International Symposium on Systems Engineering (ISSE)*. IEEE.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 82–88. AAAI Press.

Friedl, J. E. (2006). *Mastering regular expressions*. O'Reilly Media, Inc.

Hartjen, L., Philipp, R., Schuldt, F., and Friedrich, B. (2020). Saturation effects in recorded maneuver data for the test of automated driving.

Hauer, F., Schmidt, T., Holzmüller, B., and Pretschner, A. (2019). Did we test all scenarios for automated and autonomous driving systems? In *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*. IEEE.

Koopman, P. and Fratrik, F. (2019). How many operational design domains, objects, and events? In *Workshop on Artificial Intelligence Safety 2019*, volume 2301 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Langner, J., Grolig, H., Otten, S., Holzäpfel, M., and Sax, E. (2019). Logical scenario derivation by clustering dynamic-length-segments extracted from real-world-driving-data. In *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems, VEHITS 2019*. SciTePress.

Menzel, T., Bagschik, G., Isensee, L., Schomburg, A., and Maurer, M. (2019). From functional to logical scenarios: detailing a keyword-based scenario description for execution in a simulation environment. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE.

Menzel, T., Bagschik, G., and Maurer, M. (2018). Scenarios for development, test and validation of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium, IV 2018*. IEEE.

Montanari, F., German, R., and Djanatliev, A. (2020). Pattern recognition for driving scenario detection in real driving data. In *IEEE Intelligent Vehicles Symposium, IV 2020*. IEEE.

Montanari, F., Ren, H., and Djanatliev, A. (2021). Scenario detection in unlabeled real driving data with a rule-based state machine supported by a recurrent neural network. In *93rd IEEE Vehicular Technology Conference, VTC 2021*. IEEE.

Mörchen, F. (2006). Time series knowledge mining.

PEGASUS Project (2019). The PEGASUS method. [Online; accessed 05-December-2021].

Pütz, A., Zlocki, A., Bock, J., and Eckstein, L. (2017). System validation of highly automated vehicles with a database of relevant traffic scenarios. *12th ITS European Congress*.

Riedmaier, S., Ponn, T., Ludwig, D., Schick, B., and Diermeyer, F. (2020). Survey on scenario-based safety assessment of automated vehicles. *IEEE Access*, 8.

Ries, L., Langner, J., Otten, S., Bach, J., and Sax, E. (2019). A driving scenario representation for scalable real-data analytics with neural networks. In *2019 IEEE Intelligent Vehicles Symposium, IV 2019*. IEEE.

Rodrigues, J., Folgado, D., Belo, D., and Gamboa, H. (2019). SSTS: A syntactic tool for pattern search on time series. *Inf. Process. Manag.*, 56(1).

Scholtes, M., Westhofen, L., Turner, L. R., Lotto, K., Schuldes, M., Weber, H., Wagener, N., Neurohr, C., Bollmann, M., Körtke, F., Hiller, J., Hoss, M., Bock, J., and Eckstein, L. (2021). 6-layer model for a structured description and categorization of urban traffic and environment. *IEEE Access*, 9.

Silberschatz, A., Korth, H. F., and Sudarshan, S. (2020). *Database System Concepts, Seventh Edition*. McGraw-Hill Book Company.

Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., and Maurer, M. (2015). Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, volume 2015-Octob. IEEE.

VDA QMC Working Group 13 / Automotive SIG (2017). Automotive SPICE Process Assessment / Reference Model.

Wirth, R. and Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. Springer-Verlag London, UK.