Support of System Identification by Knowledge Graph-Based Information Fusion

András Földvári
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Hungary
fandras95@gmail.com

András Pataricza
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Hungary
pataric@mit.bme.hu

***Abstract*—The paper presents a knowledge graph-based solution for creating the core models for supervisory control of complex Cyber-Physical Systems (CPS) and computing infrastructures from design models and operational logs.**

The core element of modern supervisory control approaches is a digital twin, which maps the observations about the system into a hybrid run-time model representing the expected system state. It serves as a basis for interaction between the controller and the controlled system.

The high-level discrete-state machine of digital twins represents the different operational regimes (domains of similar behavior) and transitions between them. A continuous model describes the intra-domain behavior in detail. A special case is the qualitative domain model using discretized state variables in the form of a few ordered values (e.g. low, medium, high). This model category is extremely beneficial, when representing partial, dimensioning dependent behavior.

Parts of the digital twin model can be directly derived from the design models, but the description of dynamics of the qualitative domain models necessitates system identification from observations (operation logs or benchmark results). This way the creation of the digital twin necessitates information fusion from different sources. Knowledge graphs provide an abstract semantic framework for this purpose.

Our goal is to support system identification by deductive reasoning performing step-by-step checks of the abstract model to assure consistency and completeness of the observations and their respective evolving models.

***Index Terms*—cyber-physical systems, system identification, knowledge graph, digital twin**

I. Introduction

The purpose of cyber-physical systems (CPS) [1] is to observe and control the physical world through intelligent mechanisms. They operate over continuous and discrete signals originating in the physical world, for which they consist of physical and computational components interacting through communication layers [2].

The core concept in modern supervisory control of CPSs is the "digital twin." Data delivered by sensors continuously synchronize this model of the system under control with the physical world. Assurance of dependability and resilience of critical CPSs necessitates the faithfulness of the twin model.

Modern CPS design relies on the integration of pre-implemented components. The compliance to the designated temporal properties (timeliness, throughput, etc.) necessitates a proper dimensioning of the resources allocated to the components prior to the deployment.

The performance domain can influence the logic behavior. Non-linear effects resulting in bottlenecks, like the saturation of a particular resource, may change the dynamic behavior of the system. Moreover, CPS activates the built-in overload protection mechanisms, this way the behavior of a particular component, subsystem, and the entire system depend both on the functional logic (functional architecture of the system) and on its parametrization.

This way, scalability of the digital twin, similar to the deployed system requires hybrid modeling (Fig. 1) approach separating the dimensioning-independent overall logic of the behavior (discrete domain) and its actual state within an operation regime under the current workload and parametrization (continuous domain).

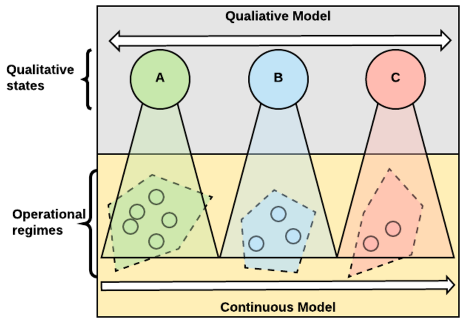

Fig. 1. Hybrid modeling

Creating a hybrid model as part of the system identification process necessitates the clustering of the data into domains (operational regimes), which show qualitatively identical behavior of the system. This task referred to as discretization can

The results presented in this research report were established in the framework of the professional community of Balatonfüred Student Research Group of BME-VIK to promote the economic development of the region. During the development of the achievements, we took into consideration the goals set by the Balatonfüred System Science Innovation Cluster and the plans of the "BME Balatonfüred Knowledge Center", supported by EFOP 4.2.1-16-2017-00021.).

be performed by an expert manually or by automated means.

Manual methods executed by an expert have the advantage that the discretization process can rely on the background knowledge of the expert. Furthermore, experts are also able to use their domain skills for variable selections, among others. On the other hand, if the number of the measured characteristics is large (as required for finite granular models), a pure expert-based approach becomes impossible.

A. *Objective*

Our goal was to support system identification (Fig. 2) by merging the prior knowledge and the qualitative system model into a knowledge graph and perform step-by-step checks of the abstract model to assure consistency and completeness of the observations and their respective evolving models.

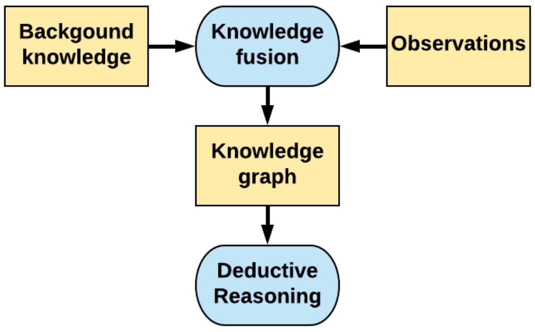

Fig. 2. Reasoning

One approach to achieve this goal uses the *phenomenological behavior* of the system which provides an abstract view of the system.

In our approach, we extend the phenomenon analysis with *prior knowledge* about the system. Knowledge graphs serve as information representation and fusion tool. Extending the knowledge graph with prior knowledge about the system provides a more detailed view and allows more precise reasoning about the system state.

If a new information (observation, input) does not fit into the knowledge graph it could indicate that 1) the digital twin model does not fit to the real system; 2) it indicates a faulty operation in the real system; or 3) the inputs are noisy. This way, deductive reasoning on the knowledge graph helps to identify these behaviors.

B. *Structure of the paper*

The rest of the paper is split into four main sections.

1) Section II presents how qualitative reasoning supports the definition of operational modes.
2) Section III presents the types of the prior knowledge in form of engineering models.
3) Section IV presents causal models in details and presents a causal model building method based on the engineering model.
4) Sections V presents a knowledge graph-based information fusion and deductive reasoning.

II. QUALITATIVE REASONING

The discrete (qualitative) state machine is an abstract form of representing the logic of the dynamic behavior of the system. Its granularity corresponds to individual operation regimes (clusters of states of similar behavior) mapped to individual states with transitions activated by crossing the inter-cluster boundaries in the continuous state space. A continuous sub-model associated with each discrete state describes the intra-cluster behavior in detail.

An upper, discrete ”super”-model assures portability independent of the actual dimensioning when it covers the union of all abstract behaviors potentially occurring in some configuration.

It allows (qualitative) reasoning [3] about the system behavior by highlighting potential phenomena at a logic level. Moreover, it allows running simulations after the parametrization of the qualitative model to a quantitative one. Although, discrete modeling has many advantages, due to the high level of abstraction it may also cause ambiguity.

Moreover, the structure of the model is typically unknown, and its creation necessitates *observation-based system identification* or *prior knowledge-based model building*. Benchmarking and operational log analysis are the primary means to ensure the match between model architecture and observations.

The model formulation is about to determine the input description of the system. Input description takes into account the knowledge of the kinds of entities and phenomena that can occur (model fragment). It is also necessary to add constraints to the model about the boundaries of the system — this collected knowledge called domain theory. Knowledge bases allow storing the domain theory by providing a rich set of functionality (e.g., built-in reasoning) and representation mechanisms (relations, attributes, rules, etc.).

A. *Clustering*

The goal of clustering is the aggregation of the fundamentally similar states into a uniform qualitative state in the discrete state machine of the digital twin representing different operational modes. There are several approaches to achieve this goal:

1) **Speculative approach:** As operational modes at least in the logic domain and runtime resource management are subjects of the design process an initial clustering can be extracted from the design models. However, due to the complex interaction between logic functionality and resource management, the initial model has to be refined on the basis of observations originating in targeted experiments, benchmarks and operational log mining.
2) **Visual methods:** For example, visual EDA uses diagrams (e.g., scatter plots, time-series diagrams) to identify cluster and their respective boundaries of each operational mode. Because it is a heuristic process, it requires comprehensive domain knowledge.

3) **Algorithmic clustering:** Algorithmic clustering (e.g., decision tree, support vector machines, random forest, k-means) partitions the observed data into blocks corresponding to uniform operation modes. These are algorithmic processes that need manual verification to check the consistency with other models.

The representativity of the inout dataset needs verification by comparing the generated set of clusters with the initial state machine for proper coverage of states and transitions in a similar way as testing of models.

a) Cluster boundaries: Mapping continuous variables into qualitative values requires the definition of thresholds for discretization by a classification algorithm according to the clusters identified:

- **Input data**: The input of the digital twin is quantitative data. The discretized data perform synchronization of the digital twin with the compatible sets of the controlled system states. Classification (into operational modes) of the continuous incoming data is based on the identified thresholds.
- **Magnitude of the actuation**: Thresholds can be used to identify the magnitude of the actuation. This way, it can provide a more precise value for actuation by transforming the qualitative values into continuous ones.

III. Background knowledge

The input information of the method is a set of observations that usually came from benchmarks or operational logs. They describe the system (output metrics) under specific workload parameters.

The first step is to analyze the measurement campaign on its own. Experts have to take into account the context of the measurement and outlier data.

The context of the measurement covers the measured parameters and boundaries of the measurement campaign. Outlier data can warn about a non-functional operation of the system or indicates that the measurement is not trustworthy.

However, different parametrizations of the same experiment (i.e., different resource allocation) may expose profoundly different phenomena. A scalable model has to merge all of this even potentially different behaviors. This way, the model building has to be adopted to the fundamental configuration settings.

A deeper understanding of the measurement data requires a priori knowledge of the domain expert.

A. Modeling approaches

Processing the measurement requires having (partially) the system architecture and functional model. The information extracted from these models can be used during the evaluation phase. The design and development phase of the system provides background information on different abstraction levels.

However, it is possible to work with partial knowledge about the system. It is not necessary to know all the details (e.g., third-party components as a black box, only the input-output parameters are known with integration details).

Analysis of a system requires the collection of all prior knowledge that is available for the analyst. The background knowledge comes from different sources and covers different aspects of the system and includes the *architecture*, *functional*, *resource allocation*, *deployment*, and *causal model* of the system.

The *system architecture* and the *functional model* provides a high abstraction about the system components and their objectives. Causal models can be built by extracting information from other engineering models.

The *resource allocation* model closely connects to the functional model. It describes which component uses which resource (e.g., networking capabilities, CPU, RAM). The installation model presents the physical or logical layout of the system.

The *causal model* expresses the causal connections in the system based on prior knowledge about the domain and the previous models. Furthermore, it is possible to extend the causal model by adding external (out of the measurement campaign's context) causal connections to the model.

Collecting and systematizing the background knowledge is necessary for further analysis.

IV. Causal model

Causality is a natural, universal concept, so deeply present in our everyday life that we instinctively think in causal relations without pondering about their actual complexity and importance. The whole physical world around us is fueled by causality. It is the connection through which -under certain circumstances- one thing (the cause) influences another (the effect) in a deterministic way.

Causal models [4] [5] allow the exploration of the causal context of a system and the detection of independent properties and events. Causal graphs are one representation of causal models.

A causal graph is a *Directed Acyclic Graph* (DAG), where the relations represent the causation among the variables. Two variables of interest are distinguishable: 1) the *exposure* (independent variable, cause); 2) and the *outcome* (dependent variable, effect). Other variables (whether measured or not measured) are called *covariates*. Covariates can be categorized into several roles and they help in the further analysis of the system.

It is possible to build causal models (Fig. 3) by using classical engineering techniques (e.g. UML, SysML [6]). Classical engineering models collect the background knowledge that is required for building the causal model. The causal model is derivable from the functional model of the system and its resource allocation model (together with the deployment instance).

The *functional model* describes the continuous processes of the system, which defines the skeleton of the causal model. The causal model uses the described data flow by the functional model.

Extending the functional model with the *resource allocation model* also extends the causal graph with detailed causal

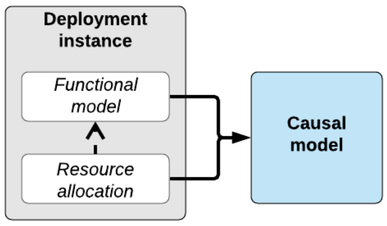

Fig. 3. Causal model building

relations. This model can present the causal connections between the resources and the functions (e.g., it is observable if two components using the same resource). This knowledge is usually verified by an expert who knows the system and its domain in great detail.

V. Knowledge representation

Knowledge graph (KG) is a kind of a database to organize complex networks of data. It provides a general-purpose approach to organize, efficiently store and query complex schemata to capture abstract concepts, entities, instances (represented as nodes of a graph) and their relations (the edges). Moreover, advanced knowledge database engines provide strong reasoning capabilities in an explainable and reusable form.

The simplicity and general validity of the underlying mathematical paradigm facilitate the use of a KG-based NoSQL database as the core element of digital twin creation and instantiation by merging the a priori knowledge with the observations.

The key asset (Fig. 4) ***in the creation phase of the digital twin model structure*** is an ontology-style merging of the design models (*architecture*, *functional*, *resource allocation*, *deployment*, and *causal model*) describing the different aspects of the system [7]. The methodology considers different input data and metamodels during the information fusion and uniformization.

A refinement of the initial core model in the KG enriches it with the domains of the variables, and their interactions as formulated in the qualitative model after processing the teaching set of observations.

Finally, the incoming stream of observations triggers a check of the consistency of the incoming data with the system model and updates the state of the digital twin model in the KG.

In this paper, GRAKN.AI [8] was used to store the models and qualitative benchmarking data.

A. Knowledge graph building

The schema of the GRAKN.AI knowledge graph is based on ER (Entity-Relationship) modeling. ER models include entities, relations, and attributes. It defines the objects of the examined world and the relationship between the objects. The objects could have attributes that describe their properties.

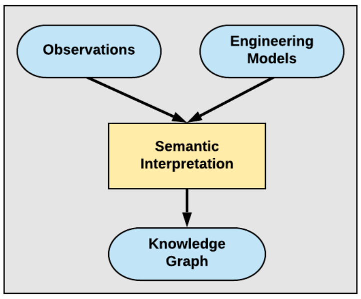

Fig. 4. Knowledge database

The language allows the definition of type hierarchies, hyper-entities, hyper-relations, and rules.

This way, it is possible to define the knowledge graph on different abstraction levels. The traceability between the abstraction levels is performed by the built-in reasoning mechanism.

B. Deductive reasoning

The knowledge graph accepts those observations which comply with the operation of the system represented by the knowledge graph. One of our research question is the following: How should we handle data that violates the operation of the system represented by the knowledge graph?

Violation can indicate different behaviors:

1) the digital twin model does not fit to the real system;
2) it indicates a faulty operation in the real system;
3) the inputs are noisy.

The identification of the violation requires further analysis involving domain experts and algorithmic mechanisms.

GRAKN.AI provides user-defined rules to support deductive reasoning. Rules look for a given pattern in the dataset and when found, create the given queryable relation. The rule-based reasoning allows automated capture and evolution of patterns within the knowledge graph.

VI. Summary and further research

Qualitative reasoning and knowledge graph management of the system models provide an abstract semantic framework for information fusion from different sources and automated model extraction. They define the operational modes of the system and makes it possible to verify the ranges by discrete value representation.

Deductive reasoning checks the compliance and completeness of the observations and their respective evolving models.

Further research is needed to generalize the models with respect to the observations. This way a general hypothesis can be constructed that is generally valid in similar operational modes. Also, if the hypothesis is proven to be valid, it will be reusable.

References

[1] A. Bondavalli, S. Bouchenak, and H. Kopetz, *Cyber-Physical Systems of Systems: Foundations–A Conceptual Model and Some Derivations: the AMADEOS Legacy*. Springer, 2016, vol. 10099.

[2] "ISO/IEC/IEEE42010 Systems and software engineering – Architecture description," International Organization for Standardization, Standard, 2011.

[3] K. D. Forbus, "Qualitative modeling," *Foundations of Artificial Intelligence*, vol. 3, pp. 361–393, 2008.

[4] J. Pearl, *Causality: models, reasoning and inference*. Springer, 2000, vol. 29.

[5] J. Pearl and D. Mackenzie, *The book of why: the new science of cause and effect*. Basic Books, 2018.

[6] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: Systems Modeling Language*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2008.

[7] A. Pataricza, L. Gönczy, A. Kövi, and Z. Szatmári, "A methodology for standards-driven metamodel fusion," in *Proceedings of the First International Conference on Model and Data Engineering*, ser. MEDI'11. Berlin, Heidelberg: Springer-Verlag, 2011, p. 270–277.

[8] "Grakn Labs Ltd: GRAKN.AI." https://grakn.ai.