Joel Greenyer, Malte Lochau, and Thomas Vogel (Hrsg.): Explainable Software for Cyber-Physical Systems (ES4CPS), GI-Dagstuhl Seminar 19023, January 2019,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019 11

Refining incomplete models through system observations with NuSMV

Mathias Blumreiter[1]

1 Motivation

Formal models have proven to be a suitable tool to tackle complexity. Hence, they are used in all engineering phases, for operation, and for documentation. However, often, the observed system behaviour does not follow the modelled behaviour. Reasons for such discrepancies may lie in inconsistencies between high and low-level models, inconsistencies between timed and discrete models, misunderstood or vague (under-specified) requirements, or just the further development of the system while the models are not kept up-to-date. For that reason, models tend to be incomplete. Cyber-physical systems (CPS), in particular, are no exception to model incompleteness, as the complexity of their tasks provides the complexity of themselves and therewith their models. Unlikely corner-cases may be easily missed in the specification and refinement phases. Since an observation of a system's behaviour is a temporal evolution of the system's state change, temporal models may be used to capture all possible evolutions over time. The benefit of using temporal models lies in their ability to describe how the system possibly arrived in a given state as well as the ability to predict what the system may do from that point on. Hence, depending on the abstraction level (in particular w.r.t. time), different kinds of system behaviours can be assessed (e.g., safety properties, probabilistic evaluations, etc.). As the model may be incomplete, it may happen that an observation has no direct counterpart in the existing model. In such a case, the existing models are not useful any more, neither for explanation nor prediction. For that reason, we tackle the problem of automatically refining incomplete models. New observations are incrementally integrated into the model as new facts without falsifying proven knowledge (i.e., without removing former observations and originally modelled behaviour). However, the objectives of explanation and prediction are in direct conflict while integrating a new observation. In case of explanation, the observation has to be integrated such that it has the best match with known behaviours to give an idea how the observation came to be. Since such an approach introduces alternative evolutions for the matched states, the integration may also create artefacts in the model and, therewith,

[1] Hamburg University of Technology, Institute for Software Systems, Am Schwarzenberg-Campus 3 (E), 21073 Hamburg, Germany, mathias.blumreiter@tuhh.de

increase the model's imprecision. On the other hand, since the aim is to use the resulting model for prediction, the finite observation has to be integrated such that the extended model continues reasonably with known (and possibly changed) behaviour after the added observation has been evaluated. Hence, prediction requires a precise model.

2 Refining incomplete models with NuSMV

As a first step to refining incomplete models of cyber-physical systems, we concentrate on the extension of discrete models and discuss a realisation in the model checker NuSMV. For that, we assume that there is a model M given that comprises the system's behaviour known from the specification phases as well as the observations that were integrated in previous integration iterations. Furthermore, we assume a set of properties Φ expressed in Computation Tree Logic (CTL) or Linear-Time Logic (LTL) that capture, for example, fundamental system behaviour or safety requirements. Since the system's observed behaviour may only be present in form of a partial observation, we assume a finite sequence of propositions τ that hold in the respective observed system states. An outline of the integration procedure is given in algorithm 1. We start with computing the meta-model $\mathcal{M}$ of all matchings between M and τ. The meta-model is constructed such that it contains all traces of M as well as the traces that result from the possible integrations of τ. Since not all of these new traces satisfy the given properties Φ, we compute an overapproximation of Φ to reduce $\mathcal{M}$ to the model of integration candidates C. Afterwards, we iteratively select the best candidate $\tau_{\text{modification}}$ (according to a given quality metric Q) and compute the extended model $\text{M}_{\text{extended}}$. Thereby, we integrate $\tau_{\text{modification}}$ such that the state space of the resulting model is changed minimally. The reason for this minimality requirement is that we use symbolic models based on a given set of variables $\mathcal{V}$ and the integration and verification cost will increase for further integration iterations when increasing $\mathcal{V}$. Hence, we prefer to stay in a minimally changed state space with the consequence that the integration candidate may not be directly realisable in it. For that reason, we continue with an execution of our model repair algorithm. The aim is to modify $\text{M}_{\text{extended}}$ such that the properties Φ hold while preserving the traces from the original model, the former observations, and the current observation. The repair algorithm mainly works on the artefacts created by previous iterations. In case the integration candidate is not realisable in the current state space, the repair algorithm has the option to increase the state space in a minimal way to integrate $\tau_{\text{modification}}$ correctly. If the integration fails even for the increased state space, the integration procedure continues with the next candidate.

3 Open questions

We described a refinement procedure for incomplete finite time-discrete models. However, depending on the use case, such models may not be ideal to capture the relevant behavioural characteristics of a particular cyber-physical system. Hence, useful abstraction levels for temporal models and corresponding logics need further investigation.

Algorithm 1 integrate(M, Φ, τ, Q)

Require: finite set of variables $\mathcal{V}$, finite model M over $\mathcal{V}$, set of CTL or LTL properties Φ over $\mathcal{V}$, (finite) sequence of satisfiable propositions $\tau = \alpha_1 \rightarrow \cdots \rightarrow \alpha_m$ over $\mathcal{V}$, and a quality metric Q

1: $\mathcal{M} \leftarrow$ matchings(M, τ)
2: $C \leftarrow \mathcal{M} \cap$ approximate$_{\forall \phi \in \Phi}(\phi)$
3: **for all** $\tau_{\text{modification}} \leftarrow$ select_best_candidate$_{\text{Q}}(C)$ **do**
4: $\quad \text{M}_{\text{extended}} \leftarrow$ integrate(M, $\tau_{\text{modification}}$)
5: $\quad$ (success, M') $\leftarrow$ repair($\text{M}_{\text{extended}}$, Φ) **constrained by** M and $\tau_{\text{modification}}$
6: $\quad$ **return** ($\top$, M') **on** success
7: **end for**
8: **return** ($\bot$, M)

Ensure: success $\Rightarrow$ M' $\models \tau \wedge \Phi$, traces(M) $\subseteq$ traces(M'), $\neg$ success $\Rightarrow$ M = M'