Towards Model Consistency between abstract and explicit Delay-Robustness in Timed Graph Transformation System

Mustafa Ghani, Holger Giese

Hasso Plattner Institute, University of Potsdam, Potsdam, Germany

Abstract

The increasing interconnectivity of embedded software systems has led to the rise of new types of Multi-Agent Systems, such as Distributed Cyber-Physical Systems, where agents synchronize by exchanging observations and local actions with remote agents. This inter-agent message passing involves (transmission, propagation, queuing, and processing) delays, which may compromise safety in safety-critical decision-making systems due to outdated information. Therefore, we proposed a methodology to derive explicit delay-robust models (resilient to δ-delays) preserving safety from safe abstract models that assume zero-delays. However, this procedure requires iterative model checking steps. In this paper, we motivate to eliminate the need for costly iterations by exploring behavioral equivalences between explicit and abstract models to define a consistency notion. This consistency facilitates the systematic transfer of verified guarantees to unverified models, effectively eliminating the need for additional model checking.

Keywords

Cyber-Physical Systems Engineering, Formal Modeling, Model Consistency

1. Introduction

The growing interconnectivity of previously isolated embedded software systems has led to the emergence of new types of Multi-Agent Systems, such as Distributed Cyber-Physical Systems (DCPSs). To maintain synchronization in such systems, agents exchange observations and local actions with remote agents. This kind of communication, defined as inter-agent message passing, involve transmission, propagation, queuing, and processing delays [1, 2]. Delays in inter-agent message passing caused by the time elapsed between agents' actions can lead to race conditions or compromise safety requirements in safety-critical systems, as decisions may be based on outdated information. Consequently, software models must clearly differentiate between local, immediate observations (occurring with zero time delay) and remote, δ-delayed observations (requiring up to a specified δ time). In [3], we introduced a methodology to enhance the robustness of zero-delay system models against δ-delays integrated in the rule-based formalism of Timed Graph Transformation Systems. As shown in Figure 1, our approach begins with a given idealized (i.e., assuming zero-delayed inter-agent message passing) safety-critical system model S_{A0}, for which safety has been verified. Based on S_{A0}, we derive a more explicit model S_{E0} by naive extension of zero to δ-delays for inter-agent message passing. We verify safety of S_{E0} by conducting model checking. If S_{E0} reveals safety violations, we repair S_{A0} and S_{E0} in the context of the robustification step (denoted in Figure 1). We proposed in [3] this robustification step as part of our methodology to handle δ-delayed messages. As a result, we obtain S_{E1} and S_{A1}. Our goal is to generate a pair of an abstract and explicit delay-robust model (such as S_{E1} and S_{A1}) to ensure different levels of abstraction.

This approach leverages the modeled information to ensure safety while avoiding unnecessary constraints on the agents' primary behavior. However, since model checking is computationally expensive [4], we aim to make this step for S_{A1} redundant. Therby, we aim to enhance the efficiency of our proposed methodology from Figure 1. To achieve this, we aim to ensure that the verified safety of S_{E1} is inherently carried over to S_{A1} by design.

Vienna'25: 17th Central European Workshop on Services and their Composition (ZEUS), February 20–21, 2025, Vienna, Austria
mustafa.ghani@hpi.de (M. Ghani)
0009-0004-6056-2125 (M. Ghani); 0000-0002-4723-730X (H. Giese)
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

S. Böhm and M. Vidgof (Eds.): 17th ZEUS Workshop, ZEUS 2025, Vienna, Austria, 20–21 February, 2025
published at http://ceur-ws.org

CEUR-WS.org/Vol-4030/paper3.pdf

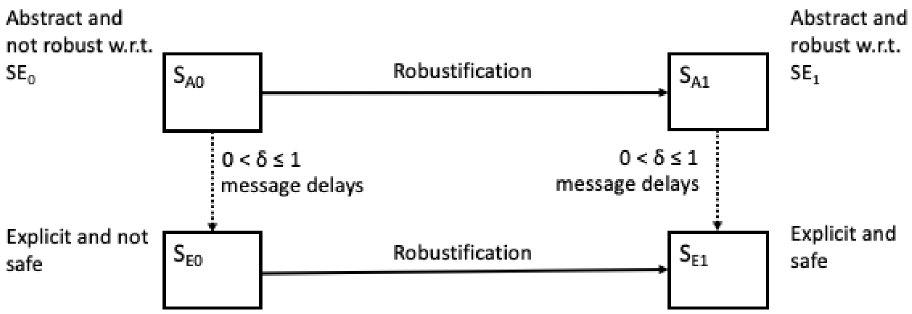

Figure 1: Methodology to derive δ-delayed-robustness for a given zero-delay system model

2. Timed Graph Transformation System

A graph G = (Gv, Ge, sG, tG) consists of a set Gv of nodes, a set Ge of edges, a source function sG: Ge $\rightarrow$ GV, and a target function tG: Ge $\rightarrow$ Gv. Let G = (Gv, Ge, sG, tG) be a graph. Let G_s = (Gv_s, Ge_s, sG_s, tG_s) be a subgraph of G, if. $Gv_s \subseteq$ Gv and $Ge_s \subseteq$ Ge. G|G_s denotes G_s is a subgraph of G. For a set X of clocks Φ(X) denotes the set of all clocks contraints ϕ generated by ϕ ::= x1 $\sim$ c | xa - xj $\sim$ c | $\phi \wedge \phi$, where $\sim \in$ {<, >, $\leq$, $\geq$}, c $\in \mathbb{N} \cup \{\infty\}$ are constants, and xa, xj $\in$ X are clocks. Let X be a set of clocks. V(X) denotes the set of all functions v: X $\rightarrow \mathbb{R}$ and is called Clock valuation. Let v: X $\rightarrow \mathbb{R}$ and X' $\subseteq$ X. Then v[X' := 0] : X $\rightarrow \mathbb{R}$ is a clock reset such that for any x $\in$ X holds if x $\in$ X', then v[X' := 0](x) = 0 else v[X' := 0](x) = v(x). Let v: X $\rightarrow \mathbb{R}$ and $\delta \in \mathbb{R}$. Then v + δ: X $\rightarrow \mathbb{R}$ is a clock increment such that for any x $\rightarrow$ X holds (v + δ)(x) = v(x)+ δ. Let v: X $\rightarrow \mathbb{R}$ and ϕ be some constraint over X. Then v $\models \phi$ denotes that v satisfies the constraint ϕ. Let v0: X $\rightarrow \mathbb{R}$ be the initial clock valuation if v0(x) = 0 for every x $\in$ X. V0(X) is the singleton set containing the unique initial clock valuation. Let H = (Hv, He, sH, tH) and G be two graphs. An injective graph morphism (short: morphism) mg: G $\rightarrow$ H is a pair of mappings mv: Gv $\rightarrow$ Hv and me: Ge $\rightarrow$ He, where mv $\circ$ sG = sH $\circ$ me and mv $\circ$ tG = tH $\circ$ me. Graph Conditions (GCs) are used to state properties on graphs requiring the presence or absence of certain subgraphs in a host graph using propositional connectives and (nested) existential quantification over graph patterns. Let TG be a distinguished graph, called type graph. A type graph has attributes connected to local variables and an attribute condition (AC) over many-sorted first-order attribute logic, which is used to specify the values for those local variables. Tg = (G, mg') is a typed graph, where G is a graph and mg' is a morphism: G $\rightarrow$ TG. Let Tg_1 = (T_1, t_1) and Tg_2 = (T_2, t_2) be two typed graphs. A typed graph morphism tgm: $Tg_1 \rightarrow Tg_2$ is a morphism mg": $T_1 \rightarrow T_2$, which is compatible with the typing functions, i.e., $t_2 \circ$ mg" = t_1. Let ρ = (L, R, K, NAC, l: K $\rightarrow$ L, r: K $\rightarrow$ R, ω, prio) be a Graph Transformation Rule (short: rule), if L (called left-hand side of rule), K (called interface graph of rule), R (called right-hand side of rule) are (typed) graphs, l and r are two (typed) morphisms, NAC is a finite set of forbidden (typed) graphs X containing L, prio: $R \rightarrow \mathbb{N}$ assigns a priority to each rule, and ω is the Application Condition (ApC) that is expressed as a graph condition. The transformation procedure defining a graph transformation approach introduced by the Double Pushout Approach [5] and is used throughout in this paper. Intuitively, the adaption of graph G can be realized by using the graph transformation rule ρ, which enforces additions and removals of subgraphs from G resulting in graph Gi, if ρ can be applied to G by satisying ApC ω for a match ma: G $\rightarrow$ Gi. Finally, we define Graph Transformations. Let GTS = (R, G, prior) be a graph transformation system, if. R is a finite set of finite rules, G is a graph, and prio: R $\rightarrow \mathbb{N}$ is a mapping assigning priorities, formulated as a natural number, to each rule. Rule $r_i \in$ R with priority $p_i \in \mathbb{N}$ suppress rule $r_j \in$ R and its priority $p_j \in \mathbb{N}$ if $p_i > p_j$. A Graph Transformation Step (short: step) is if Rule ρ transforms Graph G into Graph J. A step is called the application of a rule. If G is transformed to J by a (possibly empty) sequence of rule applications/ steps, then we write G $\xrightarrow{*}$ J. Let tGTS be a Timed Graph Transformation System, then tGTS = (R, G, time, prio, NAC) is a typed timed graph transformation system (short: TGTS), if. R is a finite set of finite rules, G is a graph, time: G $\rightharpoonup \mathbb{R}_0^+$ is a partial function that maps a graph to an element of the set of all real numbers greater or equal to 0, i.e., a total timepoint, and prio: $R \to \mathbb{N}$. Note, function CN(G) = {n | n ∈ Gv ∧ mg'v(n) = Clock} identifies in every graph the nodes used for time measurement.

3. Research Objective

To bypass model checking of S_{A1} and ease the general proposed methodology, we aim to transfer verified safeness from S_{E1} to S_{A1} by design. The underlying idea to archieve this, is to explore the formal relationship between the two models (i.e., S_{E1} and S_{A1}) to leverage model consistency, which enables transferring safety. Therefore, we define the following research questions.

- Are S_{E1} and S_{A1} formally in relation?
- How can model-based guarantees be systematically transferred from S_{E1} to S_{A1} by design?

To address this research gap, we aim to identify potential behavioral equivalences among S_{E1} and S_{A1}. Establishing such a formal relation may facilitate the transfer of safety guarantees from S_{E1} to S_{A1} by design, thereby eliminating the need for model checking of the latter. However, this approach presents challenges in determining the appropriate level of abstraction required. Furthermore, S_{E1} may introduce potential states that violate safety, which were not reachable in S_{E0}.

4. Related Work

Since model-based consistency research is inherently tied to its domain, and approaches that formally reason about consistency assume additional information about what is being analyzed with respect to the consistency notion [6], we restrict ourselves to models of Timed Graph Transformation Systems with a focus on delay-robustness for mission-critical systems. In [7] the authors presented a different version of Timed Graph Transformation Systems neither supporting quantitative analysis nor considering delay-robustness. In [8, 9], inter-agent message delays were not explicit considered since message passing was restricted by allowing communication within a given timing intervall. In [3], we presented an approach to derive explicit delay-robustness for a given abstract model. However, this approach requires the verification of every generated model (i.e., S_{E0}, S_{E1}, S_{A1}) while in this work we propose a consistency relation making model checking for S_{A1} not required and assuring delay-robustness per design.

5. Conclusion and Future Work

In this paper, we discussed the motivation for reducing the computational cost and the number of model-checking iterations in our previously proposed approach by defining a consistency relation between S_{E1} and S_{A1}. Such a formal relation could serve as the foundation for systematically transferring model-based guarantees. In this context, we identified key research questions and the associated challenges related to achieving this objective.

Acknowledgments

This research was funded by the HPI Research School on Systems Design.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] B. A. Forouzan, Data communications and networking, Huga Media, 2007.
[2] M. Van Steen, A. S. Tanenbaum, A brief introduction to distributed systems, Computing 98 (2016) 967–1009.
[3] M. Ghani, S. Schneider, M. Maximova, H. Giese, Deriving delay-robust timed graph transformation system models, in: International Conference on Graph Transformation, Springer, 2024, pp. 158–179.
[4] M. Schmalz, H. Völzer, D. Varacca, Model checking almost all paths can be less expensive than checking all paths, in: International Conference on Foundations of Software Technology and Theoretical Computer Science, Springer, 2007, pp. 532–543.
[5] E. Hartmut, E. Karsten, P. Ulrike, T. Gabriele, Fundamentals of algebraic graph transformation, Monographs in theoretical computer science. An EATCS series. Springer (2006).
[6] R. Pascual, B. Beckert, M. Ulbrich, M. Kirsten, W. Pfeifer, Formal foundations of consistency in model-driven development, in: International Symposium on Leveraging Applications of Formal Methods, Springer, 2024, pp. 178–200.
[7] S. Gyapay, R. Heckel, D. Varró, Graph transformation with time: Causality and logical clocks, in: Graph Transformation: First International Conference, ICGT 2002 Barcelona, Spain, October 7–12, 2002 Proceedings 1, Springer, 2002, pp. 120–134.
[8] M. Maximova, H. Giese, C. Krause, Probabilistic timed graph transformation systems, in: Graph Transformation: 10th International Conference, ICGT 2017, Held as Part of STAF 2017, Marburg, Germany, July 18-19, 2017, Proceedings 10, Springer, 2017, pp. 159–175.
[9] M. Maximova, S. Schneider, H. Giese, Compositional analysis of probabilistic timed graph transformation systems, Formal Aspects of Computing 35 (2023) 1–79.