Verified Traffic Networks: Component-based Verification of Cyber-Physical Flow Systems

Andreas Müller
andreas.mueller@jku.at
Department of Cooperative Information System
Johannes Kepler University
4040 Linz, Austria

Stefan Mitsch
smitsch@cs.cmu.edu
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA-15213, USA

André Platzer
aplatzer@cs.cmu.edu
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA-15213, USA

***Abstract*—We address the problem how high-fidelity verification results about the hybrid systems dynamics of cyber-physical flow systems can be provided at the scale of large (traffic) networks without prohibitive analytic cost. We propose the use of contracts for traffic flow components concisely capturing the conditions for a safe operation in the context of a traffic network. This reduces the analysis of flows in the full traffic network to simple arithmetic checks of the local compatibility of the traffic component contracts, while retaining higher-fidelity correctness guarantees of the global hybrid systems models that inherits from correct contracts of the hybrid system components. We evaluate our approach in a case study of a modular traffic network and a prototypical implementation in a model-based analysis and design tool for traffic flow networks.**

I. Introduction

Cyber-physical systems (CPS) are today pervasively embedded into our lives and increasingly act in close proximity as well as with direct impact to humans. A typical example is *traffic control*: traffic operators monitor large road networks, in which *continuous physical characteristics* (e. g., traffic flow, travel times) are influenced by *discrete control measures* (e. g., speed limits). The discrete and continuous parts of road traffic are inseparably intertwined and mutually influence each other (e. g., a speed limit influences traffic flow, which in turn determines the chosen speed limit). Thus, they need to be modeled and analyzed in conjunction using so-called *hybrid system* models. Moreover, we need to analyze traffic flow at the scale of entire networks to account for effect propagation.

Traffic operators rely on software when selecting appropriate control measures to resolve potentially critical traffic situations. However, when traffic control software decides autonomously or suggests critical actions to an operator, the safety of these actions is crucial (e. g., expected traffic volume should not overload a detour). Formal verification techniques for hybrid systems allow us to analyze road traffic control and physics in conjunction to *mathematically prove correctness* for all of the infinitely many possible states—not just by analyzing some examples, as in testing or simulation. Formal verification methods based on *model checking* and *reachability analysis* focus on full automation and are therefore restricted to less expressive classes of CPS. In contrast, we consider *deductive verification techniques* which rely on human guidance, but allow for more realistic models (e. g., safety proofs for adaptive cruise control in cars [1] and for vehicle-to-infrastructure communication [2]). However, hybrid system models of entire traffic networks are huge and hard to verify.

In this paper, we attempt to solve the problem how *high-fidelity verification results* (i. e., mathematical proof) about the hybrid systems dynamics of *cyber-physical traffic flow systems* can be provided at the scale of *large traffic networks* without prohibitive analytic cost. Handling a CPS model of a full traffic network is currently hopeless without exploiting its structure during a proof. The basic observation making mathematical proof possible is that traffic networks, while large, are composed of many instances of only a small number of similar patterns (e. g., 4-way intersections, onramps, traffic lights). Component-based modeling approaches exploit this characteristic to decompose a traffic network model into smaller parts. We present a way to ensure that proofs about traffic components transfer to the network level, which is nontrivial since the components influence each other and can thus not be analyzed solely in isolation.

Our key conceptual insight is that flow-based CPS can be modeled with components having *contracts*[1] on flow inputs and flow outputs, with flow being directed by a controller. We define contracts for components and composition operations related to their load and capacity, which are important characteristics of traffic systems. When we check contracts statically during modeling, we can conclude safety of load and capacity of networks from the verification results of their components. Checking the preconditions of verified contracts on real-time traffic data makes it possible to predict whether safe or unsafe behavior is to be expected in downstream flow components. Our technical contribution is to show that safety properties of the components (e. g., their capacity is such that a certain amount of traffic can be handled) transfer to safety of the whole network. The hybrid systems verification results that we present for a library of important traffic patterns require human guidance, but the proofs only need to be done once. The checks for correct composition of the contracts that are required at the network level reduce to simple arithmetic checks, which can be performed at scale with marginal computational cost. That way components modeled and verified by hybrid system experts at design-time, can be composed and parametrized by traffic operators at run-time.

We evaluate our approach by introducing a library of traffic

[1]Contracts are a classical concept from programming languages used to understand preconditions and postconditions of programs, which we generalize to input and output flow conditions of traffic components here.

components to build modular traffic network models and by implementing it in a component-based model analysis and design tool for traffic networks. The tool checks for correct composition of the higher-level composite model to ensure that the presented proofs about its components reunite to a large safety proof about the network.

II. Related Work

We discuss related work in component-based modeling, traffic modeling and traffic management. We briefly describe each approach before we summarize comparisons to our work.

A. Component-based CPS Modeling and Verification

The most closely related approach is proposed by Damm et al. [3], who present a component-based design framework for controllers of hybrid systems focused on propagation of alarms when stability or safety constraints are violated. However, they are restricted to alarm propagation and use Lyapunov functions to ensure system stability. Simko et al. [4] propose a modeling formalism for the physical fragment of CPS components and their electrical or mechanical entanglement. Ringert et al. [5] model the discrete behavior of CPS as Component and Connector (C&C) architectures using automata. Although models of single components can be verified, they provide no guarantees about verified compositions. Peter and Givargis [6] describe components through behavior, interfaces, and meta information. Their method does not focus on inferring system correctness from verified component safety.

A field closely related to component-based verification is assume-guarantee reasoning (AGR), which was originally intended as a device to counteract the state explosion problem in model checking by decomposing a verification task into subtasks. In AGR, individual components are analyzed together with *assumptions* about their context and *guarantees* about their behavior (i. e., a component's "contract"). Various approaches (e. g., [7], [8]) have been proposed, but are often limited to linear dynamics, abstract away continuity or rely on reachability analysis, over-approximation and model checking.

In summary, only few component-based approaches handle generic CPS with both discrete and continuous aspects. The few existing ones do not focus on the impact on formal verification and theorem proving to reason about local correctness properties of components that together contribute to global system correctness properties. Furthermore, few approaches consider the additional properties that are necessary to ensure a verified composition of verified components and the consequences of component-based modeling for proofs.

B. Traffic Models

Theoretical considerations about traffic and traffic models (e. g., [9]) have been used for various tasks like transportation planning [10] or simulation [11]. Models from various fields, like swarm dymanics (e. g., [12]), or queuing theory (e. g., [13]) can be found in the literature. Traffic models are often distinguished into microscopic- and macroscopic models (e. g., [10], [12]). Microscopic models deal with the behavior of single cars and usually comprise position and velocity of individual entities. In contrast, in macroscopic models the behavior of individual vehicles is ignored in favor of sizable aggregates of many vehicles. In other words, macroscopic models consider traffic as an effectively one-dimensional compressible fluid, which inspired the flow components in this paper.

First beginnings of *macroscopic* traffic flow analysis on highways are derived from photographic observations as described by Greenshields who analyzed flow, density and velocity of highway traffic [9]. Lighthill and Whitham [14] proposed a flow model based on *kinematic waves* in long rivers, stating that there exists a relation between the flow and density within a fluid. Together with the work of Richards [15] who proposed a similar traffic flow theory, this led to the *LWR traffic model*. Lebacque et al. [16] introduced a natural extension of the LWR model with behavioral attributes of individual cars and drivers, encompassing many other macroscopic models.

Daganzo [17] and Yperman et al. [18] split a traffic network into components. Unlike our symbolic and hybrid approach, they use numerical calculations to compute flows and loads throughout the network for discrete time steps.

Loos et al. [1] verified that cars equipped with automatic cruise control will never crash, using a hybrid microscopic model of a set of individual cars. We however, focus on macroscopic models, which allow predictions beyond individual cars to a flow of multiple cars.

Traffic modeling is quite related and a plethora of different models can be found in literature. Although traffic models have become more and more sophisticated (e. g., using partial differential equations) most are solely continuous and do not consider verification. However, especially their hybrid properties play an important role in traffic control (e. g., flow influenced by traffic lights), which is why we propose a verification method for hybrid traffic models with a macroscopic flow-based continuous part and a discrete controller.

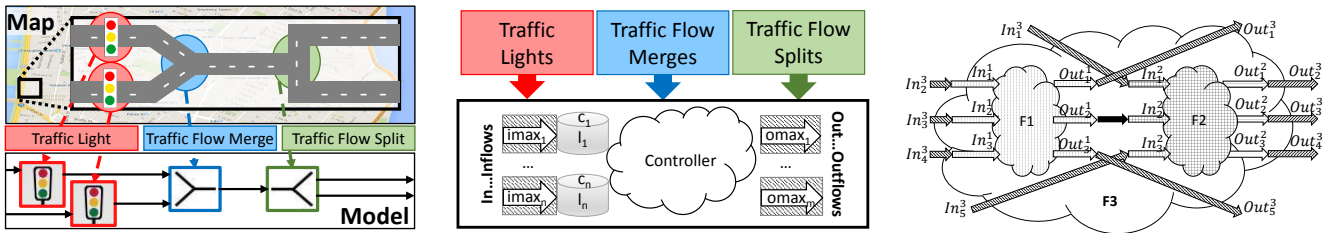
C. Intelligent Traffic Management Systems

Systems assisting operators in managing road traffic come from different research areas like situation-awareness or knowledge management. Situation-awareness approaches (e. g., [19]) try to predict the evolution of the traffic situation by projection of the evolution of single traffic objects (e. g., traffic jams) and their interactions. Knowledge management systems (e. g., [20]) support operators in their decision process by providing (situation-dependent) expert knowledge. Our approach of using component-based hybrid system models of traffic networks to reason about their safety properties acts as a complementary tool to provide evidence on the correctness of such systems, as verification had not been addressed so far.

III. Traffic Control

Traffic networks are safety-critical CPS, in which motion of cars along roads is continuous, while the traffic control is discrete (e. g., traffic lights change their state discretely). Hybrid system models of traffic networks can be used for various purposes, such as calculating approximate travel times, network planning or prediction of traffic breakdowns[2]. We focus on macroscopic traffic models for large networks, as used

[2] A traffic breakdown occurs when traffic load (i. e., queue in a link) exceeds capacity (i. e., maximum queue size) [21]. Variations of exact definitions of *traffic breakdown* exist in the literature (e. g., [22, 23]).



(a) Component Extraction (b) Component Construction (c) Component Composition

Figure 1: Cyber-Physical Traffic Flow Components

by traffic operators in control centers to monitor and influence road traffic. Poor decisions in traffic centers can imply serious consequences for road traffic: e. g., congestion caused by a misplaced road-block can propagate to a highway, where large speed differences increase the risk of accidents.

Traffic management systems intend to support operators by suggesting actions, which presumably entail desirable results (e. g., congestion dissolves) and avoid undesirable side effects (e. g., traffic breakdowns). The goal in this paper is to check for a potential *traffic breakdown*. We consider a traffic breakdown as any kind of congestion that is not restricted to a single part of the road network, but propagates through it, which happens as soon as the load of cars in front of any intersection (i. e., the queue in a link) exceeds the road's capacity (i. e., maximum queue size). Consequently, the safety property that we want to verify is that no road will exceed its capacity.

Unfortunately, due to the complex structure of larger traffic networks, the respective hybrid system models become large, consisting of innumerable roads and intersections, which all have to be checked for potential breakdowns. However, typical traffic networks are formed from only a small number of traffic control patterns, such as traffic lights or intersections. Fig. 1a shows how to extract sample patterns from a traffic network (we focus on traffic lights, split and merge, similar to [17] and [18]). Their structural similarity makes them amenable to component-based modeling. Thus, we represent each pattern as a traffic component with a separate hybrid system model (cf. Fig. 1b) consisting of: (i) one or more roads where vehicles can enter the component (i. e., In), (ii) one or more roads where vehicles can exit the component (i. e., Out), (iii) the maximum number of vehicles that can pass through the road in a given time (i. e., $i_{\max}$, number of vehicles per time unit), which is usually restricted by physical characteristics (e. g., number of lanes or speed limits), (iv) the maximum number of vehicles that pass through the output in a given time (i. e., $o_{\max}$, number of vehicles per time unit), (v) the actual number of vehicles present on the corresponding input at some point in time (i. e., current queue in link l) and (vi) the number of vehicles allowed on an inflow without exceeding the infrastructural limits (i. e., maximum queue size c, number of vehicles), which usually depends on environmental characteristics (e. g., a long stretch of road with several lanes vs. a short single lane). In order to build the entire network from these components, we need a composition operation (cf. Fig. 1c). Sequential composition of two traffic components represents consecutive parts of the traffic network: cars leaving the first traffic component move on as inputs to the second traffic component.

As we want to avoid traffic breakdowns, we still have to show that load never exceeds the capacity throughout the traffic network. Although this is impractical for large networks, it can be done for single components. However, we have to make sure that the properties of single traffic components transfer to their composition. If this is guaranteed, it suffices to decompose a traffic network into components, verify that no breakdown occurs in each component, rebuild the network using the safe components and obtain an overall breakdown-free model. In Section IV we formally define the foundations for such a component-based modeling and verification approach.

IV. Component-Based Flow Verification

This paper considers traffic network properties abstractly as capacity, load and flow. As a main safety property, we are interested in proving that, given certain initial conditions and bounds, the network does not exceed its capacity for a designated time. In this section, we define flow components and their properties. Then we define composition operations to connect flow components to form composite flow components which retain the properties of their individual parts.

A. Flow Components

We model (traffic) networks as directed graphs G of vertices V and edges E. The concrete behavior of a flow component is described by its hybrid systems model (Section V). Its abstract role in the traffic network is merely a vertex (cf. Fig. 1b), which takes flow from a set of *input* edges and directs it to a set of *output* edges, each having a *maximum flow*. The inputs furthermore have a *capacity* and a *load* which is influenced by the in- and outflows. A controller in the flow component decides how flow from the inputs is directed to outputs. Def. 1 gives a definition of flow components according to the pattern described in Section III.

Definition 1 (Flow Component): Let E be the set of all edges. A *flow component* F is defined as a tuple

$$F = (\text{In}, \text{Out}, i_{\max}, o_{\max}, l, c) \text{ where}$$

- $\text{In} \subseteq E$ is a finite ordered set $\{\text{In}_1, \ldots, \text{In}_n\}$ of n input names.
- $\text{Out} \subseteq E$ is a finite ordered set $\{\text{Out}_1, \ldots, \text{Out}_m\}$ of m output names.
- $i_{\max} : \text{In} \to \mathbb{R}^+$ is a function assigning a non-negative maximum inflow to each input in In. We lift to ordered sets as follows $i_{\max}(\text{In}) = \{i_{\max}(\text{In}_1), \ldots, i_{\max}(\text{In}_n)\}$.

- $o_{\max} : \text{Out} \to \mathbb{R}^+$ is a function assigning a non-negative maximum outflow to each output in Out. We lift to ordered sets as follows $o_{\max}(\text{Out}) = \{o_{\max}(\text{Out}_1), \ldots, o_{\max}(\text{Out}_m)\}$.
- $c : \text{In} \to \mathbb{R}^+$ is a function assigning a maximum capacity (i. e., maximum manageable load) to each input in In. We lift to ordered sets $c(\text{In}) = \{c(\text{In}_1), \ldots, c(\text{In}_n)\}$.
- $l : \left(\text{In}, \mathbb{R}^+, \mathbb{R}^+, (\mathbb{R}^+)^m\right) \to \mathbb{R}^+$ is a function calculating the load (i. e., capacity used) of an input depending on the current time, the inflow $i_{\max}$ and all outflows $o_{\max}$.

B. Safe Flow Components

Here, we define *contracts* that must hold to ensure that a flow component does not exceed the capacity of its inputs. These contracts are conditions on the maximum inflows, outflows, capacities and loads of a component, as well as on the time that has passed. Outputs of a component do not need a capacity, because they are included in the input capacities of subsequent components. The actual flow to and from a component can vary between 0 and the provided maximum flow values. Safe uses of a traffic flow component need to ensure that, within a given time horizon, the load on any input never exceeds the capacity of the input (cf. Def. 2).

Definition 2 (Load Safety): A flow component is *load-safe* for time t, if and only if its load l does not exceed the capacity c of any of its inputs until time t:

$$\forall \text{In}_i \in \text{In} \;.\; l\big(\text{In}_i, t, i_{\max}(\text{In}_i), o_{\max}(\text{Out})\big) \leq c(\text{In}_i) \quad .$$

C. Safe Sequential Composition of Flow Components

After defining safe components, we want to compose them in a way that remains safe. A sequential composition of two flow components links one or more outputs of the predecessor component to the same number of inputs of the successor component (cf. Fig. 1c and Def. 3). We focus on loop-free composition, since cyclic flows are undesirable in road traffic. When there are loops present in a traffic network, we focus on paths between a source and a destination point (e. g., the beginning and end of an area of road work) and analyze the resulting sequential paths (cf. Fig. 1a).

Definition 3 (Sequential Composition): Let

$$F^s = (\text{In}^s, \text{Out}^s, i^s_{\max}, o^s_{\max}, l^s, c^s)\,, \text{ for } s \in \{1, 2\}$$

be flow-components, with disjoint inputs and outputs (i. e., $\text{In}^1 \cap \text{In}^2 = \text{Out}^1 \cap \text{Out}^2 = \emptyset$) and $\mathcal{C} : \text{Out}^1 \rightharpoonup \text{In}^2$ be a partial (i. e., not every output must be mapped when connecting two components), injective (i. e., every input is only mapped to one output upon connection of components) function, mapping connected outputs and inputs between the two components. We define $\mathcal{O}$ as the domain of $\mathcal{C}$ (i. e., all values $x \in \text{Out}^1$ such that $\mathcal{C}(x)$ is defined) and $\mathcal{I}$ as the the image of $\mathcal{C}$ (i. e., all values $y \in \text{In}^2$ such that $y = \mathcal{C}(x)$ holds for some $x \in \text{Out}^1$).

We define the sequential composition $F^3 = F^1 \bullet\!\!-\!\!\circ^{\mathcal{C}} F^2$ of flow components F^1 and F^2 by connecting outputs of F^1 to inputs of F^2 according to a function $\mathcal{C}$, with $|\mathcal{O}| > 0$, where

$$F^3 = \left(\text{In}^3, \text{Out}^3, i^3_{\max}, o^3_{\max}, l^3, c^3\right) \text{ with}$$

- $\text{In}^3 = (\text{In}^2 \setminus \mathcal{I}) \cup \text{In}^1$
- $\text{Out}^3 = \text{Out}^2 \cup (\text{Out}^1 \setminus \mathcal{O})$
- $n_3 = \left|\text{In}^3\right| = \left|\text{In}^1\right| + \left|\text{In}^2\right| - |\mathcal{C}|$ and $m_3 = \left|\text{Out}^3\right| = \left|\text{Out}^1\right| + \left|\text{Out}^2\right| - |\mathcal{C}|$
- $i^3_{\max} : \text{In} \to \mathbb{R}^+$, with $\forall \text{In}_k \in \text{In}^1 \;.\; i^3_{\max}(\text{In}_k) = i^1_{\max}(\text{In}_k)$ and $\forall \text{In}_l \in \text{In}^2 \cap \text{In}^3 \;.\; i^3_{\max}(\text{In}_l) = i^2_{\max}(\text{In}_l)$
- $o^3_{\max} : \text{Out} \to \mathbb{R}^+$, with $\forall \text{Out}_k \in \text{Out}^1 \cap \text{Out}^3 \;.\; o^3_{\max}(\text{Out}_k) = o^1_{\max}(\text{Out}_k)$ and $\forall \text{Out}_l \in \text{Out}^2 \;.\; o^3_{\max}(\text{Out}_l) = o^2_{\max}(\text{Out}_l)$,
- $l^3 : \left(\text{In}, \mathbb{R}^+, \mathbb{R}^+, (\mathbb{R}^+)^{m_3}\right) \to \mathbb{R}^+$, with
$$\forall \text{In}_k \in \text{In}^1 \;.\; l^3\left(\text{In}_k, t, i^3_{\max}(\text{In}_k), o^3_{\max}(\text{Out}^1)\right) = l^1\left(\text{In}_k, t, i^3_{\max}(\text{In}_k), o^3_{\max}(\text{Out}^1)\right) \text{ and}$$
$$\forall \text{In}_l \in \text{In}^2 \cap \text{In}^3 \;.\; l^3\left(\text{In}_l, t, i^2_{\max}(\text{In}_l), o^3_{\max}\left(\text{Out}^2\right)\right) = l^2\left(\text{In}_l, t, i^2_{\max}(\text{In}_l), o^3_{\max}\left(\text{Out}^2\right)\right)$$
- $c^3 : \text{In} \to \mathbb{R}^+$, with $\forall \text{In}_k \in \text{In}^1 \;.\; c^3(\text{In}_k) = c^1(\text{In}_k)$ and $\forall \text{In}_l \in \text{In}^2 \cap \text{In}^3 \;.\; c^3(\text{In}_l) = c^2(\text{In}_l)$.

The composition of two flow components is not necessarily again a flow component: A composite flow component further needs a guarantee that the connected—and thus hidden—inputs of the second flow component do not overflow.

Definition 4 (Composite Flow Component): The sequential composition $F^1 \bullet\!\!-\!\!\circ^{\mathcal{C}} F^2$ (Def. 3) of two flow components F^1 and F^2 is called a flow component, if and only if

$$\forall x \in \mathcal{O} \;.\; y = \mathcal{C}(x) \to l^2\Big(y, t, o^1_{\max}(x), o^2_{\max}\left(\text{Out}^2\right)\Big) \leq c^2(y)\,.$$

Load-safety for composite flow components is defined as in Def. 2. An easier sufficient (even if not necessary) criterion for a composite to be load-safe is presented in Theorem 1.

Theorem 1 (Load-safe composition): Let F^1 and F^2 be flow components, $F^3 = F^1 \bullet\!\!-\!\!\circ^{\mathcal{C}} F^2$ their sequential composition and t any point in time. Then F^3 is a load-safe flow component for time t, if

1) F^1 and F^2 are load-safe at time t and
2) $\forall x \in \mathcal{O} \;.\; y = \mathcal{C}(x) \to o^1_{\max}(x) \leq i^2_{\max}(y)$.

Theorem 1 (proof available online[3]) implies that it is sufficient to check $o^1_{\max}(\mathcal{O}) \leq i^2_{\max}(\mathcal{I})$ for all connected pairs of ports to ensure that a composition of load-safe components is load-safe. This is crucial, as it provides a purely arithmetic condition that can easily be checked by a modeling tool at run-time, even for a large number of components.

V. Verified Traffic Flow Component Library

To show the applicability of our work, we develop a library of traffic components consisting of (i) a traffic light, (ii) a traffic flow merge component, which merges two roads into one road and (iii) a traffic flow split component, which splits a road into two roads, and model and verify that their

[3] http://www.tk.jku.at/people/mueller/publications/itsc15/

high-fidelity hybrid systems dynamics satisfies their safe flow contracts. These components can already be used to model a good deal of actual road networks, as they can be combined to form larger components (e. g., n-way merges and splits). Fig. 1a shows a snippet of a road map, modeled using only these three traffic components.

To allow modeling of traffic with a flow model, we assume that all vehicles are of uniform size and similar behavior. We do not yet account for changes in driver behavior due to changes in flow (e. g., during congestion), but we consider changes in route preferences through nondeterministic flow ratios in components with multiple outflows. Note that, while we consider an arbitrary positive inflow bound by $i_{\max}$, we assume that *outflow* from a component is maximal (i. e., $o_{\max}$ as long as $l > 0$ and equal to the current inflow otherwise), since we expect drivers want to pass through the network as fast as possible. We map the members of a flow-component F to a part of the road network as proposed in Section III.

As modeling formalism for the internal dynamics of traffic flow components we use differential dynamic logic (dL), which is a first-order dynamic logic for hybrid programs, a program notation for hybrid systems [24]. Such dL models can be verified using the tool KeYmaera [25], which is open source and has been applied for verification of several case studies[4]. We provide the models and proofs for the traffic components online[3], and integrated as projects into KeYmaera.

A. Traffic Light Component

We define a traffic light as a stretch of road, having two states, namely *red* and *green* (the *orange* state is ignored here, since it is just a heads-up for red). If the traffic light is red, no flow along the road is possible and vehicles start to queue up at the traffic light. Upon switching to green, the vehicles start to flow away from the traffic light as fast as possible. This represents a simplification of the actual behavior of vehicles in front of a traffic light, which usually brake and accelerate slowly, resulting in so called shock-waves [15]. We consider a basic traffic light with equally long red and green cycles and with parametric cycle length $T_c \in \mathbb{R}^+$. Extension with different phases and timings is possible, but not discussed here.

A traffic light component has a single inflow ($n = 1$) and a single outflow ($m = 1$), which are separated through a traffic light. The hybrid program of the traffic flow at a traffic light *tl* with a constant cycle length T_c can be found in Model 1.

Model 1 Traffic flow in a traffic light

$$tl \equiv (ctrl_{tl}; plant_{tl})^* \quad (1)$$

$$ctrl_{tl} \equiv \text{if } (t_c = T_c) \text{ then } t_c := 0;\ go := (go-1)^2 \text{ fi} \quad (2)$$

$$i_{\text{act}} := *;\ ?(0 \le i_{\text{act}} \le i_{\max}); \quad (3)$$

$$\text{if } (l > 0) \text{ then } o_{\text{act}} := o_{\max} \quad (4)$$

$$\text{else } o_{\text{act}} := \min(i_{\text{act}}, o_{\max}) \text{ fi}; \quad (5)$$

$$plant_{tl} \equiv l' = i_{\text{act}} - o_{\text{act}} \cdot go, t' = 1, t_c' = 1 \quad (6)$$

$$\&\ t_c \le T_c \wedge l \ge 0 \quad (7)$$

[4]c.f. http://symbolaris.com/info/KeYmaera.html

In Model 1, *go* is the status of the traffic light (i. e., red is 0, green is 1), T_c is the length of a red- and a green-cycle and t_c is the timer measuring the cycle length. The variables l, $i_{\max}$ and $o_{\max}$ represent the load, maximum inflow, and maximum outflow as per Def. 1, while i_{act} and o_{act} are the actual flows. The model consists of a discrete controller and a continuous plant, which are repeated a non-deterministic number of times (1). In (2) we check whether or not the timer measuring the cycle length has been running for exactly T_c units of time, (i. e., the traffic light status has not changed within the last T_c units of time). If so, we reset the timer t_c and toggle the traffic light status in (2). In (3) we set the actual inflow non-deterministically to be between 0 and $i_{\max}$. In (4) and (5) we set the actual outflow depending on the current load. Outflow is set to maximum if the load is positive; otherwise the outflow is determined by the inflow, but still restricted by $o_{\max}$. The continuous evolution in (6) and (7) linearly increases time t and the traffic light timer t_c, and runs at most for T_c units of time and while the load l is non-negative. Furthermore, it updates the load l by actual flows i_{act} and o_{act} depending on the traffic light status *go*.

Proposition 1 (Traffic Light Load Safety): We want the traffic light to be load-safe in order to avoid an overflow which would result in a traffic breakdown. A flow component with one input and one output is load-safe per Def. 2 if

$$l\left(\text{In}_1, t, i_{\max}\left(\text{In}_1\right), \{o_{\max}\left(\text{Out}_1\right)\}\right) \le c\left(\text{In}_1\right) \ .$$

Thus, a traffic light is safe (ψ_{tl}) if it is load-safe for up to a maximum time T.

$$\psi_{tl} \equiv (t \le T \rightarrow l \le c)$$

When started in a safe initial state ϕ_{tl}, the traffic light component *tl* ensures load safety ψ_{tl}

$$\phi_{tl} \rightarrow [tl]\psi_{tl} \quad (8)$$

where

$$\begin{aligned}\phi_{tl} \equiv\ & t = 0 \wedge 0 \le t_c \le T_c \wedge T_c > 0 \wedge T > 0 \wedge l = 0 \\ & \wedge c \ge \max(T_c \cdot i_{\max}, T \cdot i_{\max} - \max\left(0, o_{\max} \cdot \frac{T - T_c}{2}\right)) \\ & \wedge 0 \le i_{\max} \wedge 0 \le o_{\max} \wedge go \cdot (go - 1) = 0 \ .\end{aligned}$$

The dL formula in (8) expresses that all runs of the traffic light model *tl*, when started in states where ϕ_{tl} is true, reach states where ψ_{tl} is true. We proved Proposition 1—i. e., dL formula (8)—using KeYmaera. The theorem prover enforces to find correct conditions ϕ_{tl} and ψ_{tl}, which are non-trivial, but necessary to successfully verify the models.

Infinite time: If we want the traffic light to be load-safe forever, we need to increase T towards infinity. As a result, we can obtain a traffic light component that is safe for infinite time when $o_{\max} \ge 2 \cdot i_{\max}$.

B. Traffic Flow Merge Component

A traffic flow merge component is defined as two roads with certain maximum inflows and capacities, which merge into one road. Accordingly a traffic flow merge component has two inflows ($n = 2$) and only one outflow ($m = 1$). In a microscopic modeling view, vehicles can only pass from one

input road through the component. With a macroscopic view, flow can merge from two inputs onto one output at the same time. Here, hybrid programs allow us to talk about both views at the same time: a discrete jump in the controller chooses the flow ratio between the two roads nondeterministically. If the controller chooses only the boundary values (0 or 1), then the model describes a microscopic per vehicle view; otherwise it describes a macroscopic flow view.

Model 2 Traffic flow in a traffic flow merge component

$$\mathit{tfm} \equiv (\mathit{ctrl}_{\mathit{tfm}};\ \mathit{plant}_{\mathit{tfm}})^* \tag{9}$$
$$\mathit{ctrl}_{\mathit{tfm}} \equiv \mathit{road} := *;\ ?(0 \leq \mathit{road} \leq 1); \tag{10}$$
$$i1_{\text{act}} := *;\ ?(0 \leq i1_{\text{act}} \leq i1_{\max}); \tag{11}$$
$$i2_{\text{act}} := *;\ ?(0 \leq i2_{\text{act}} \leq i2_{\max}); \tag{12}$$
$$\text{if } (\mathit{l1} > 0 \lor \mathit{l2} > 0) \text{ then } o_{\text{act}} := o_{\max} \tag{13}$$
$$\text{else } o_{\text{act}} := \min(i1_{\text{act}} + i2_{\text{act}}, o_{\max}) \text{ fi}; \tag{14}$$
$$\mathit{plant}_{\mathit{tfm}} \equiv \mathit{l1}' = i1_{\text{act}} - o_{\text{act}} \cdot (1 - \mathit{road}), t' = 1, \tag{15}$$
$$\mathit{l2}' = i2_{\text{act}} - o_{\text{act}} \cdot \mathit{road}\ \&\ \mathit{l1} \geq 0 \land \mathit{l2} \geq 0 \tag{16}$$

In Model 2 *road* selects the ratio of flow directed from inflows to the outflow. The variables *l1*, *l2*, $i1_{\max}$, $i2_{\max}$ and $o_{\max}$ represent the loads, maximum inflows, and maximum outflow as per Def. 1. In (10) the flow ratio is changed. If $\mathit{road} = 0$, all flow from the first input is directed to the output while the second input is blocked, if $\mathit{road} = 1$, all flow comes from the second input. In (11) to (14) we again set the actual flows. The continuous evolution in (15) and (16) increases time t linearly and updates the loads *l1* and *l2* by actual flows $i1_{\text{act}}$, $i2_{\text{act}}$ and o_{act} depending on the current road distribution.

Proposition 2 (Merge Load Safety): We want the traffic flow merge component to be load-safe in order to avoid an overflow which would result in a traffic breakdown. A flow component with two inputs and one output is load safe if

$$l\left(\text{In}_i, t, i_{\max}\left(\text{In}_i\right), \{o_{\max}\left(\text{Out}_1\right)\}\right) \leq c\left(\text{In}_i\right) \text{ for } i \in \{1, 2\}\ .$$

Thus, a traffic flow merge is safe (ψ_{tfm}) if it is load-safe for up to a maximum time T:

$$\psi_{\mathit{tfm}} \equiv \left(t \leq T \rightarrow (\mathit{l1} \leq c1 \land \mathit{l2} \leq c2)\right)\ .$$

A traffic flow merge component *tfm* ensures load safety ψ_{tfm}, cf. (17), when started in a safe initial state ϕ_{tfm} (18).

$$\phi_{\mathit{tfm}} \rightarrow [\mathit{tfm}]\psi_{\mathit{tfm}} \tag{17}$$
$$\begin{aligned}\phi_{\mathit{tfm}} \equiv\ & t = 0 \land 0 \leq i1_{\max} \land 0 \leq i2_{\max} \land 0 \leq o_{\max} \\ & \land c1 \geq T \cdot i1_{\max} \land c2 \geq T \cdot i2_{\max} \\ & \land \mathit{l1} = \mathit{l2} = 0 \land 0 \leq \mathit{road} \leq 1\end{aligned} \tag{18}$$

We used KeYmaera to verify that Proposition 2—i. e., dL formula (17)—holds for the merge component of Model 2.

Infinite time: Infinite load safety for a traffic flow merge component cannot be guaranteed for Model 2, because only one single road could be active all the time. Thus, the other road would require infinite capacity to be load-safe. Infinite-time load safety requires a fairness condition in Model 2.

Fairness: Currently, each input of the traffic flow merge component has to assume that the other input is active all the time, which in turn means that none of them actually ever assumes any outflow. To overcome this issues, one can introduce a fairness criterion by means of a fairness factor X that indicates the minimum rate of the outflow that is guaranteed for each of the inputs. That way, each input can assume at least some outflow. Additional information about introducing fairness criteria into the traffic flow merge component is available online[3].

C. Traffic Flow Split Component

A traffic flow split component is defined as a road having a certain maximum inflow and a capacity, which splits into two roads. Accordingly a traffic flow split component has one inflow ($n = 1$) and two outflows ($m = 2$). Akin to the merge component, a discrete jump in the controller chooses the flow ratio between the two roads nondeterministically.

Model 3 Traffic flow in a traffic flow split component

$$\mathit{tfs} \equiv (\mathit{ctrl}; \mathit{plant})^* \tag{19}$$
$$\mathit{ctrl}_{\mathit{tfs}} \equiv i_{\text{act}} := *;\ ?(0 \leq i_{\text{act}} \leq i_{\max}); \tag{20}$$
$$\mathit{road} := *;\ ?(0 \leq \mathit{road} \leq 1); \tag{21}$$
$$\text{if } (l > 0) \text{ then } o1_{\text{act}} := o1_{\max}; o2_{\text{act}} := o2_{\max} \tag{22}$$
$$\text{else } o1_{\text{act}} := \min(i_{\text{act}}, o1_{\max}); \tag{23}$$
$$o2_{\text{act}} := \min(i_{\text{act}}, o2_{\max}) \text{ fi}; \tag{24}$$
$$\mathit{plant}_{\mathit{tfs}} \equiv l' = i_{\text{act}} - o1_{\text{act}} \cdot (1 - \mathit{road}) - o2_{\text{act}} \cdot \mathit{road}, \tag{25}$$
$$t' = 1\ \&\ l \geq 0 \tag{26}$$

In Model 3, l, $i_{\max}$, $o1_{\max}$ and $o2_{\max}$ represent the load, maximum inflow, and maximum outflows as per Def. 1. The actual inflow is set in (20). The variable *road* selects the flow distribution to the outflows and is changed in (21). In a sense, it models expected driver preferences: If $\mathit{road} = 0$ then all flow is directed to the first output, if $\mathit{road} = 1$ to the second output. Then, in (22)–(24) the outflows are adjusted depending on the load. For example, consider a highway exit where drivers prefer to exit instead of following the mainline, so that a queue builds up on the mainline upstream from the exit ($l > 0$). The model captures the exit preference with a choice of *road* close to one. Even though in this case the mainline downstream of the exit could have $o1_{\max}$ outflow in theory, it will only produce $o1_{\max} \cdot (1 - \mathit{road})$ flow, cf. (25). When there is no queue on the mainline ($l \leq 0$), in (23) to (24) we simply distribute the inflow (again obeying the maximum possible outflows). The continuous evolution in (25) and (26) linearly increases time t and updates the load l by actual flows i_{act}, $o1_{\text{act}}$ and $o2_{\text{act}}$ depending on the flow ratio.

Proposition 3 (Split Load Safety): We want traffic flow split components to be load-safe in order to avoid an overflow which would result in a traffic breakdown. A flow component with one input and two outputs is load-safe per Def. 2 if

$$l\left(\text{In}_1, t, i_{\max}\left(\text{In}_1\right), \{o_{\max}\left(\text{Out}_1\right), o_{\max}\left(\text{Out}_2\right)\}\right) \leq c\left(\text{In}_1\right)\ .$$

Thus, a traffic flow split component is safe ψ_{tfs} if it is load-safe

for up to a maximum time T.

$$\psi_{tfs} \equiv (t \leq T \rightarrow l \leq c)$$

When started in a safe initial state ϕ_{tfs}, the traffic flow split component *tfs* ensures load safety ψ_{tfs}

$$\phi_{tfs} \rightarrow [tfs]\psi_{tfs} \tag{27}$$

where

$$\begin{aligned}\phi_{tfs} \equiv\ & t = 0 \wedge T > 0 \wedge 0 \leq i_{\max} \wedge 0 \leq o1_{\max} \wedge 0 \leq o2_{\max} \\ & \wedge\, c \geq \max\Big(0, T \cdot \big(i_{\max} - \min\left(o1_{\max}, o2_{\max}\right)\big)\Big) \\ & \wedge\, l = 0 \wedge 0 \leq \mathit{road} \leq 1 \ .\end{aligned}$$

We used KeYmaera to verify that Proposition 3—i. e., dℒ formula (27)—holds for the component in Model 3.

Infinite time: If we want the intersection to be load-safe forever, we need to increase T towards infinity. This way we can obtain that a traffic flow split component is safe for infinite time, if $o1_{\max} \geq i_{\max} \wedge o2_{\max} \geq i_{\max}$.

VI. SAFE-T Modeling Tool

To evaluate our traffic flow component library, we implemented a prototypical modeling tool (SAFE-T, i. e., Safety Analysis Flow-component Editor for Traffic networks[3]) for traffic network configuration. SAFE-T lets traffic operators instantiate and compose traffic components, and automatically check load-safety of the resulting network using the criteria of Propositions 1–3. The respective properties (e. g., capacities, flows) can be set according to environmental features (e. g., number of lanes) and traffic situations (e. g., flows differ from rush hour to off-peak times).

In order to form a traffic network from our traffic component library, the modeling tool uses the sequential composition operation as defined in Section IV-C. It checks the arithmetic conditions (cf. Theorem 1) and highlights connections where the maximum outflow would have been too large with a "(!)". Additionally, since sequential composition of two traffic components means that the components represent consecutive parts of the traffic network, the tool sets the flows of connected In and Out to the minimum of the respective $i_{\max}$ and $o_{\max}$. Because SAFE-T checks this and all traffic components presented above are proven load-safe in KeYmaera, we know that their composition in the whole network is again load-safe.

The toolbar at the top of the tool provides access to the component library (cf. V) to form a *network graph* (i. e., traffic light, merge and split). A dedicated start component without inputs supplies constant flow on a single output, whereas an end component without outputs consumes flow. The *time slider* at the bottom of the screen sets the current time t used in calculations. When a component is selected, the left part of the work area visualizes input load over time in a *load graph* per input, and allows adaption of the *component properties* (e. g., name, flows). For each input, the tool prints the loads wrt. the selected time and capacities above the respective components.

Furthermore, the tool allows analysis of *spillback* (i. e., overflow propagation) within the network, where we assume that as soon as an overflow in a component occurs (i. e., as soon as the load reaches the capacity, or to put it another way,

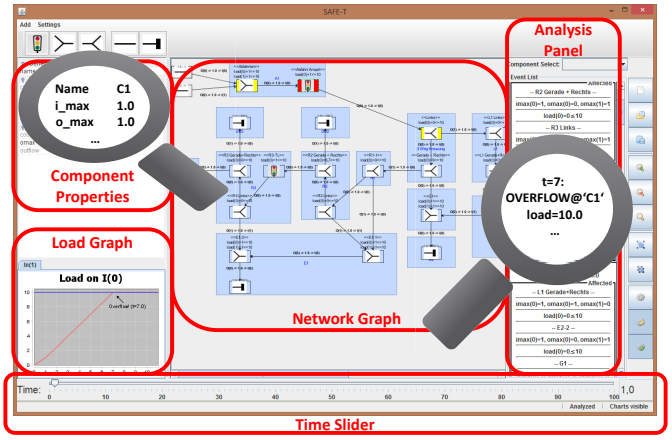


Figure 2: Screenshot with illustrations, showing a SAFE-T traffic model of an actual highway exit in Austria

as soon as the queue is no longer limited to the component at hand, but spills back to the preceding component) all inflow and outflow of this component is reduced to zero, thus changing the behavior of connected components and propagating the overflow through the network. The *analysis panel* contains a list of overflow events, ordered ascendingly by time. When selecting a list entry, the respective component is highlighted red and all components affected by the overflow (i. e., all connected components, as their flows to respectively from the overflown component are reduced to zero) become yellow. This way, the user can not only see at which point in time an overflow occurs, but also how this overflow propagates through the network and reaches a point of interest (e. g., when does a traffic jam caused by a road block reach the highway).

The example in Fig. 2 (tool and examples available online[3]) shows the model of a highway exit in Austria, created using only components presented in this paper (and additional start/end components). By moving the time slider, one can see how the load within the different components changes or which component produces an overflow. On the right hand side, the analysis panel is displayed and the selected traffic light component, producing an overflow after 7 units of time for the configuration at hand, is highlighted in red. This leads to a reduction of flow to zero in the two connected components, which are highlighted in yellow. The advantage of our approach is that the safety of the overall system can be derived from properties of the single components, because the behavior of the components, as well as their compositions are verified. Also the flow values can be adapted depending on current events (e. g., rush hour) by changing certain parameters, without remodeling the entire network.

VII. Conclusion and Future Work

We presented a component-based modeling and verification approach for (traffic) flow in (road) networks that can be used to reduce verification complexity of CPS models. The approach uses two modeling levels: (i) high-fidelity flow models, which define the hybrid system behavior of flow components, and (ii) higher-level specifications to build a large network from provably correct components. We further presented the

modeling tool SAFE-T for composition of verified traffic flow components. SAFE-T supports automatic checking of the derived arithmetic conditions to ensure safety of the overall traffic network and to analyze the propagation of overflows. The tool can be (i) enhanced with further flow components, (ii) adapted for different flow-based CPS application areas and (iii) extended to check further constraints (e. g., actual flow).

Our current approach is a first step towards a proof-aware model composition approach, focused on flow-based CPS considering maximum flow values and simplified continuous dynamics. This paves the way for multiple further research directions: (i) extend component definitions with actual flow, (ii) extend models to consider traffic phenomena (e. g., shock-waves), (iii) consider models with loops, (iv) include further composition operators (e. g., delayed flow propagation), (v) introduce further traffic components (e. g., multi-lane merge/split), (vi) explore methods to automatically transform road network graphs into components and compositions, and (vii) generalize to non-flow types of components.

ACKNOWLEDGEMENTS

Work partly funded by BMVIT under grant FFG BRIDGE 838526, by REA PIOF-GA-2012-328378, by US DOT UTC TSET grant, award# DTRT12GUTC11, by FWF P28187-N31, and by OeAD Marietta Blau grant ICM-2014-08600.

REFERENCES

[1] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," in *FM*, ser. LNCS, M. Butler and W. Schulte, Eds., vol. 6664. Springer, 2011, pp. 42–56.

[2] S. Mitsch, S. M. Loos, and A. Platzer, "Towards formal verification of freeway traffic control," in *ICCPS*. IEEE, 2012, pp. 171–180.

[3] W. Damm, H. Dierks, J. Oehlerking, and A. Pnueli, "Towards component based design of hybrid systems: Safety and stability," in *Time for Verification*, ser. LNCS, Z. Manna and D. Peled, Eds. Springer, 2010, vol. 6200, pp. 96–143.

[4] G. Simko, T. Levendovszky, M. Maroti, and J. Sztipanovits, "Towards a Theory for Cyber-physical Systems Modeling," in *Proc. of the 4th ACM SIGBED Int. Workshop on Design, Modeling, and Evaluation of CPS*. New York, NY, USA: ACM, 2014, pp. 56–61.

[5] J. O. Ringert, B. Rumpe, and Wortmann A., "From Software Architecture Structure and Behavior Modeling to Implementations of Cyber-Physical Systems," in *Software Engineering 2013 Workshopband*, Köllen Druck+Verlag GmbH, Ed., 2013, pp. 155–170.

[6] S. Peter and T. Givargis, "Utilizing Intervals in Component-based Design of Cyber Physical Systems," in *IEEE 16th Int. Conf. on Comp. Sci. and Eng. (CSE), 2013*, J. Chen, Ed. IEEE, 2013, pp. 635–642.

[7] L. Benvenuti, D. Bresolin, P. Collins, A. Ferrari, L. Geretti, and T. Villa, "Assume–guarantee verification of nonlinear hybrid systems with Ariadne," *Int. J. of Robust and Nonlinear Control*, vol. 24, no. 4, pp. 699–724, 2014.

[8] T. A. Henzinger, M. Minea, and V. Prabhu, "Assume-guarantee reasoning for hierarchical hybrid systems," in *HSCC*, ser. LNCS, M. Di Benedetto and A. Sangiovanni-Vincentelli, Eds. Springer, 2001, vol. 2034, pp. 275–290.

[9] B. D. Greenshields, J. T. Thompson, H. C. Dickinson, and R. S. Swinton, "The Photographic Method Of Studying Traffic Behavior," *Highway Research Board Proceedings*, vol. 13, 1934.

[10] H. Rakha and B. Crowther, "Comparison of Greenshields, Pipes, and Van Aerde Car-Following and Traffic Stream Models," *Transportation Research Record: J. of the Transportation Research Board*, vol. 1802, pp. 248–262, 2002.

[11] D. Sun, L. Zhang, and F. Chen, "Comparative study on simulation performances of CORSIM and VISSIM for urban street network," *Simulation Modelling Practice and Theory*, vol. 37, pp. 18–29, 2013.

[12] N. Bellomo and C. Dogbe, "On the modeling of traffic and crowds: A survey of models, speculations, and perspectives," *SIAM Review*, vol. 53, no. 3, pp. 409–463, 2011.

[13] J. M. Smith, "Application of State-Dependent Queues to Pedestrian/Vehicular Network Design," *Operations Research*, vol. 42, no. 3, pp. 414–427, 1994.

[14] M. J. Lighthill and G. B. Whitham, "On kinematic waves. II. A theory of traffic flow on long crowded roads," *Proc. R. Soc. A*, vol. 229, no. 1178, pp. 317–345, 1955.

[15] P. I. Richards, "Shock Waves on the Highway," *Operations Research*, vol. 4, no. 1, pp. 42–51, 1956.

[16] J. P. Lebacque, S. Mammar, and H. H. Salem, "Generic Second Order Traffic Flow Modelling," in *Transportation and Traffic Theory*, R. Allsop, M. Bell, and B. Heydecker, Eds. Elsevier, 2007, pp. 755–776.

[17] C. F. Daganzo, "The cell transmission model, part II: Network traffic," *Transportation Research Part B: Methodological*, vol. 29, no. 2, pp. 79–93, 1995.

[18] I. Yperman, S. Logghe, and B. Immers, "The Link Transmission Model: an efficient implementation of the kinematic wave theory in traffic networks," in *Proceedings of the 10th EWGT Meeting*, 2005.

[19] N. Baumgartner, S. Mitsch, A. Müller, A. Salfinger, W. Retschitzegger, and W. Schwinger, "A tour of BeAware: A situation awareness framework for control centers," *Information Fusion*, vol. 20, pp. 155–173, 2014.

[20] K. Almejalli, K. Dahal, and M. A. Hossain, "Intelligent Traffic Control Decision Support System," in *Applications of Evolutionary Computing*, ser. LNCS, M. Giacobini, Ed. Springer, 2007, vol. 4448, pp. 688–701.

[21] H. Wang, K. Rudy, J. Li, and D. Ni, "Calculation of traffic flow breakdown probability to optimize link throughput," *Appl. Math. Model.*, vol. 34, no. 11, pp. 3376–3389, 2010.

[22] J. L. Evans, L. Elefteriadou, and N. Gautam, "Probability of breakdown at freeway merges using Markov chains," *Transportation Research Part B: Methodological*, vol. 35, no. 3, pp. 237–254, 2001.

[23] B. Persaud, S. Yagar, and R. Brownlee, "Exploration Of The Breakdown Phenomenon In Freeway Traffic," *Transportation Research Record: J. of the Transportation Research Board*, vol. 1634, pp. p. 64–69, 1998.

[24] A. Platzer, "Differential dynamic logic for hybrid systems." *J. Autom. Reas.*, vol. 41, no. 2, pp. 143–189, 2008.

[25] A. Platzer and J.-D. Quesel, "KeYmaera: A hybrid theorem prover for hybrid systems." in *IJCAR*, ser. LNCS, A. Armando, P. Baumgartner, and G. Dowek, Eds., vol. 5195. Springer, 2008, pp. 171–178.