Exploring Weakly-hard Paradigm for Networked Systems

Chao Huang
Northwestern University
Evanston, Illinois
chao.huang@northwestern.edu

Kacper Wardega
Boston University
Boston, Massachusetts
ktw@bu.edu

Wenchao Li
Boston University
Boston, Massachusetts
wenchao@bu.edu

Qi Zhu
Northwestern University
Evanston, Illinois
qzhu@northwestern.edu

ABSTRACT

Networked systems have shown great promises in various cyber-physical applications, such as automotive and transportation systems, smart buildings and infrastructures, and robotic systems. As these systems employ advanced components and interact closely with the dynamic environment, they are often subject to significant disturbances from environment interference, security attacks, and device faults. To ensure system safety, performance and other properties, it is critical to capture these disturbances and reason about their impact at the network level. In this work, we propose to use weakly-hard constraints to specify the disturbances in a bounded manner, and leverage them to formally reason about system properties. We will first present two case studies that demonstrate the impact of disturbances on various properties in networked systems and motivate the usage of weakly-hard constraints. We will then discuss several possible research directions in applying weakly-hard constraints to networked systems.

CCS CONCEPTS

• **Networks → Network performance evaluation**; • **Computer systems organization →** *Embedded and cyber-physical systems*; *Dependable and fault-tolerant systems and networks*;

1 INTRODUCTION

With the rapid growth of cyber-physical and Internet-of-Things (IoT) applications, engineering systems have become increasingly networked and distributed. These networked systems often employ complex computation and communication components, and interact closely with dynamic and uncertain environment. During their operation, substantial disturbances may occur due to environment interference, security attacks, or device faults. Such disturbances could affect sensing, computation, communication, storage, or actuation operations, and ultimately lead to incorrect system behavior or degraded performance. For instance, connected vehicles applications based on vehicular ad-hoc networks have shown great promises in improving transportation safety and efficiency. In these applications, vehicles exchange information with each other and the surrounding infrastructure via wireless vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) messages. Such wireless communication may be subject to long communication delays or packet losses due to environment noise or security attacks (e.g., jamming or flooding attacks), which could lead to unsafe scenarios and inferior performance [57, 59, 60]. For such networked systems, it is crucial to formally capture the possible disturbances to different operations and analyze their impact on various system properties.

There have been a large number of works in addressing disturbance to networked systems in the literature. The disturbances may come from permanent faults, transient disturbances, or intermittent errors. For tolerating permanent node faults, researchers have tried to either provision spare resources in the design phase [22, 26, 43], or modify the network when runtime disturbances occur [1, 4, 9, 10]. For handling transient disturbances of networks, both centralized scheduling policies [11, 40, 62] and distributed ones [52, 53] have been proposed. To manage the intermittent errors of communication in networked control systems (e.g., packet losses), researchers have modeled them as stochastic processes [47, 49, 50, 54] and explore the relation between the likelihood of packet loss and the concerned functional property such as stability.

In this paper, we consider leveraging weakly-hard constraints to capture the disturbances on various operations (e.g., sensing faults, computation deadline misses, communication delays or losses). The notion of weakly-hard constraints was initially proposed for specifying timing requirements in real-time systems. While the traditional hard deadlines require every instance of task execution to complete within its deadline, weakly-hard constraints allow occasional deadline misses in a bounded manner. A common example is the (m,K) constraints, which specify that at most m instances of a task are allowed to violate their execution deadlines among any K consecutive instances [6, 21, 36]. Compared with hard deadlines, such weakly-hard constraints more accurately capture the timing requirements for system functions (many of which can in fact tolerate certain degrees of deadline misses), and significantly improve system feasibility and flexibility. Compared with soft deadlines, where any deadline miss is allowed, weakly-hard constraints provide deterministic guarantees on system safety, stability, performance, data quality and other properties.

In the literature, weakly-hard constraints have been mostly applied to address computation deadline misses on a single node. We believe that weakly-hard paradigm could also be a powerful tool for modeling, analyzing, designing, and validating networked systems, on two major aspects. First, it provides a bounded specification for disturbances on various operations, and thus facilitates formal and more deterministic reasoning of system properties. Deriving weakly-hard constraints for those operations may not be trivial, but as the operations are typically local (e.g., sensing and computation faults or deadline misses on a single node, communication delays or packet losses on a single link), bounding their behavior is often easier and more practical than directly analyzing system properties at the network level. From this perspective, specifying weakly-hard constraints can be viewed as a way to provide guarantees on network-level properties by imposing the requirements on lower-level operations. Such requirements can be used to guide the design of lower-level components, and to monitor the lower-level operations at runtime for early prediction of system failures or assurance of system properties under disturbances. It is also worth noting that the (m, K) weakly-hard constraints provide a unified formulation for capturing the disturbance. That is, permanent faults can be viewed as a special case with $m = K$, while a single fault can be viewed as $m = 1$ within a certain K window.

Second, weakly-hard constraints can be leveraged to allow "controlled skipping" of operations when needed, e.g., skipping (not executing) certain sensing, computation and communication operations, as long as the weakly-hard constraints are satisfied and hence the system properties and requirements are still met. By skipping those operations, the system may utilize the saved resources for other objectives, e.g., running security monitoring or fault tolerance tasks; or it may simply go to idle state to reduce energy consumption.

In the rest of the paper, we will first present two case studies in Section 2 that discuss the disturbance in networked systems and motivate the usage of weakly-hard constraints. In particular, Section 2.1 shows an example of network flooding application under node faults defined by weakly-hard constraints. Section 2.2 discusses how connected vehicle applications can be affected by communication disturbances, and how such disturbances may be captured with a partial consensus model and weakly-hard constraints. Section 3 then presents some of the relevant approaches for these problems, and highlights the advantages for using weakly-hard constraints. We will then discuss some of the possible research directions on using weakly-hard paradigm for networked systems in Section 4 and conclude the paper in Section 5.

2 MOTIVATING EXAMPLES

2.1 Network Flooding

A key advantage of weakly-hard models for networks is the ability to obtain non-probabilistic guarantees, as one might with a hard real-time model. We substantiate this claim with the ubiquitous example of network flooding. The aim of a network flooding algorithm is to distribute a packet to every node on a network from a single, arbitrary source node. The standard example of a flooding algorithm is to simply send any incoming packet to every neighbor except for the neighbor from which the packet was originally received. Typically, flooding algorithms prioritize minimizing the time required for every node to receive the packet over minimizing network traffic.

For simplicity, consider a network $G = (V, E)$, where V denotes a set of nodes and $E \subseteq V^2$, operating in synchronous stages under a real-time model where a node will send a received packet to each of its neighbors in every time step. In this model, missing a deadline is equivalent to failing to propagate the packet at the corresponding time step. Clearly the maximum latency, i.e. the first time at which the last node receives the packet, is equal to the diameter of G when there are no deadline misses (the diameter of G, called D_G, is the maximum distance between any pair of nodes in G). When deadline misses occur, the maximum latency is unbounded under a probabilistic model since it is always possible (albeit with a vanishingly small probability) that a node misses all its deadlines in any finite window. By adopting a weakly-hard model we are able to dispense with probabilistic models and obtain an exact maximum latency. In the case of synchronous flooding, this can be achieved using the following SAT/SMT formulation.

Given the network G, we first obtain the adjacency matrix $(a_{i,j})$. We assume that each node has a known (m, K) constraint – this means that for every consecutive K steps of the algorithm, each node may be turned off or refuse connection or miss the communication deadline at most m times. We introduce symbolic values $(p_{i,t})$ to signify that node i has the packet at time t and $(m_{i,t})$ to signify that node i has turned off at time t. To model the operation of the flooding algorithm for finite horizon T, we use pseudo-Boolean constraints:

$$\mathtt{WH} := \bigwedge_i \bigwedge_t \sum_t^{\min(t+K,T)} m_{i,t} \leq m \tag{1}$$

which ensures that each node is off for no more than m timesteps for every consecutive K-length period,

$$\mathtt{EVOLVE} := \bigwedge_{i,j} \bigwedge_{t<T-1} p_{i,t} \wedge \neg m_{i,t} \wedge \neg m_{j,t} \wedge a_{i,j} \implies p_{j,t+1} \tag{2}$$

which details how packets move between neighbors that are both on at a given timestep,

$$\mathtt{PERSIST} := \bigwedge_i \bigwedge_t p_{i,t} \implies p_{i,t+1} \tag{3}$$

which posits that a packet that has been received at a given node persists on that node,

$$\mathtt{INIT} := \sum_i p_{i,0} = 1 \tag{4}$$

which guarantees the existence of a unique node to act as the source of the flood, and finally

$$\mathtt{FLOOD} := \bigwedge_i \bigvee_t p_{i,t} \tag{5}$$

which is the clause that defines a successful run of the flooding algorithm where each node eventually obtains the packet.

We combine these clauses to obtain

$$\phi(T) := \neg\mathtt{FLOOD} \wedge \mathtt{INIT} \wedge \mathtt{EVOLVE} \wedge \mathtt{PERSIST} \wedge \mathtt{WH} \tag{6}$$

If $\phi(T)$ is satisfiable, it means that there is a model $\mathcal{M}$ for $\phi(T)$ where a particular node in V initiates a flood and a particular pattern of deadline misses across the network causes the flooding algorithm to not succeed within T timesteps. If, on the other hand, $\phi(T)$ is unsatisfiable, it means that under the specified (m, K) constraints, any initializing node in V will be able to flood a packet to every other node in the network in at most T steps, regardless the particular pattern of deadline misses may be. By initializing T to $m + 1$ and incrementing T by one each time $\phi(T)$ is satisfiable, we obtain a guaranteed maximum latency for the simple flooding algorithm on G in the weakly-hard regime.

We experiment on a total of 860 random graphs ranging from sparsely to densely connected with small sizes $|V| \in \{10, 12, 14, 16, 18\}$. An example of a medium-dense connected graph from our test set is shown in Figure 1. We display the average per-graph increase in T relative to D_G over a variety of (m,K) constraints in Table 1. Entries marked as "-" are scenarios that are trivially impossible to flood as all neighbors can be off as long as the source node is on and vice versa.

Table 1: Average increase in maximum latency under various (m,K) constraints relative to D_G.

		K						
		3	4	5	6	7	8	9
m	1	2.59x	2.59x	2.00x	2.00x	2.00x	2.00x	2.00x
	2	-	-	4.17x	4.17x	4.17x	3.00x	3.00x
	3	-	-	-	-	5.76x	5.76x	5.76x
	4	-	-	-	-	-	-	7.35x

For a given graph G and fixed K, the maximum latency T of the synchronous flooding algorithm is monotonically increasing in m when G is under the (m,K) constraint. Furthermore, the maximum latency under fixed m is monotonically decreasing in K. These monotonic properties are intuitive results. Less intuitive is the fact that a ratio of $\frac{m}{K}$ is not enough to predict the maximum latency. As an example, the maximum latency increase for $(m,K) = (1,3)$ is 2.59x whereas for $(m,K) = (2,8)$ the maximum latency increase is 3.00x. One would intuitively expect that in the latter case, where each node can be off for a smaller fraction of time than in the former, that the maximum latency increase would be smaller. The reason for this discrepancy is that permitting longer lengths of *consecutive* per-node downtime allows for the possibility that packets are "quarantined" behind low-degree nodes of the graph. In the $(m,K) = (1,3)$ case, each node can be off for at most one consecutive timestep and so the maximum latency is smaller than the $(2,8)$ case where nodes can potentially be off for two consecutive timesteps.

2.2 Connected Vehicles

Future vehicles equipped with wireless communication modules are highly promising in bringing significant improvement in transportation safety and efficiency. These vehicles exchange messages with each other and with surrounding infrastructure via vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication channels, to inform others about their current states and intents for *collaboratively* making decisions. For instance, in cooperative adaptive cruise control (CACC), consecutive vehicles communicate with each other to maintain a safe distance between them. In centralized autonomous intersections, vehicles approaching an intersection communicate with an intersection manager that decides the order for them to pass.

However, as V2V and V2I communication may suffer from environment interference or malicious security attacks (e.g., jamming or flooding attacks) on the wireless channels, there could be substantial communication delays and losses. In our prior work [56, 57, 59], we have shown that such communication delays/losses may have

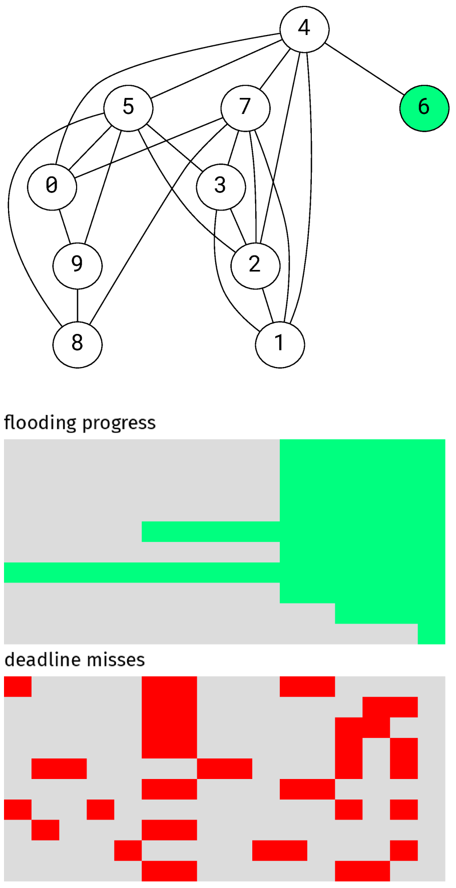


Figure 1: (top) A medium-dense graph G with 10 nodes from our test set; $D_G = 4$. In the worst case, the flood is initialized by node 6, shown in green. (bottom) Worst-case flooding latency pattern for G with $(m,K) = (2,5)$. The progress of the packet through the network is shown in green, and the corresponding deadline misses are shown in red (rows correspond to the nodes and columns correspond to timesteps).

significant impact on the performance and safety of connected vehicle applications such as CACC and autonomous intersections. For CACC, when following vehicle does not receive the V2V messages from proceeding vehicle in time, it will revert to regular adaptive cruise control (ACC), which determines the safe distance without full information of the proceeding vehicle (in particular its acceleration) [59]. For autonomous intersections, we developed a delay-tolerant intersection management protocol that explicitly addresses V2I message delays/losses to guarantee system safety, i.e., vehicles with conflicting routes will never enter the intersection at the same time [56, 57]. The protocol also ensures deadlock-free and liveness properties [1], if the maximum communication delay

[1] In this application, liveness means that every vehicle that sends request will eventually cross the intersection.

(considering resend in the case of message loss) is bounded and the bound is known. The work also quantitatively analyzes the impact of communication delays on autonomous intersection performance, and demonstrates the improvement from our approach over traditional traffic lights [57] and smart traffic lights that leverage back pressure scheduling [56], as long as the communication delay is within a reasonable bound (500ms to 1000ms).

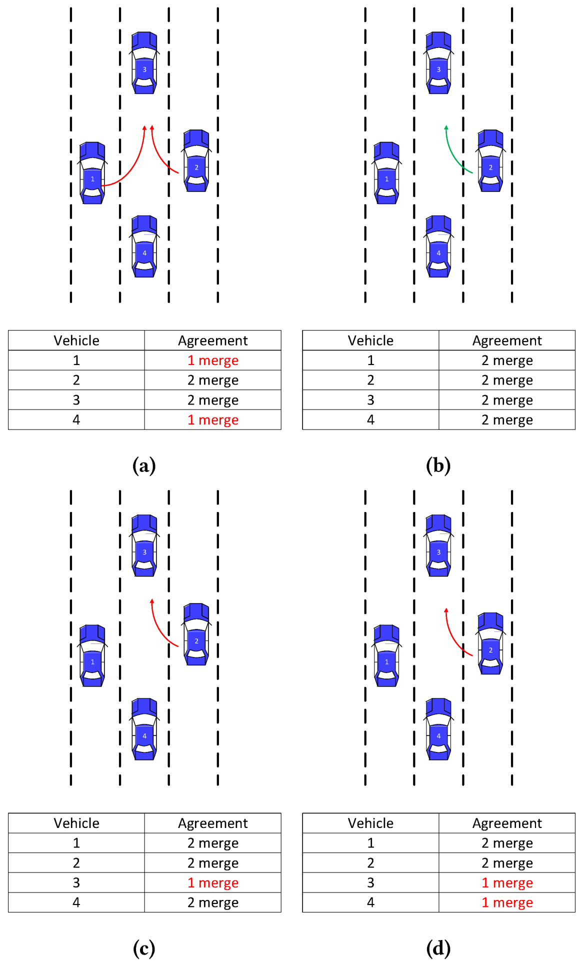
The studies on CACC and autonomous intersections demonstrate the various impacts of communication disturbance on system properties. In both works, system performance is significantly affected by message delays/losses, while system safety can always be guaranteed. Cases for deadlock-free and liveness properties are more complex – they can be ensured but only when the communication delay between any vehicle and the intersection manager is bounded (i.e., it will eventually happen), and very importantly, the protocol is designed to wait for such communication to complete. These observations motivated us to consider the following general questions for connected vehicle applications:

- For quantitative metrics such as performance, how to measure the impact of communication disturbance on them?
- For logical properties such as safety, liveness, and deadlock-free, how to derive the requirements on communication delay and reliability that can ensure the properties are met?
- For both types of analysis above, what types of formalism are needed for capturing the requirements on communication?

In [29], we find that these problems can be considered from the viewpoint of the consensus problem, as the traffic participants (vehicles and infrastructures) have to reach certain level of agreement via communication to ensure the desired system properties. The interesting observation there is that in many cases, the participants only need to reach what we call *partial consensus* under communication disturbances to meet the functionality requirements, albeit with possible performance degradation. An illustrating example on the collaborative lane merging/changing is shown in [29] and explained below.

In Figure 2, four vehicles communicate with each other and try to reach an agreement on a lane merging decision, i.e., whether vehicle 1 or vehicle 2 would merge into the central lane. In Figure 2 (a), there is no global consensus, possibly due to communication disturbances from environment interference, security attacks, or faulty devices. There is partial agreement between vehicles 1 and 4 that vehicle 1 will merge into the central lane, and between vehicles 2 and 3 that vehicle 2 will merge into the central lane. In this case, it would be unsafe to perform the lane merging, as vehicle 1 and vehicle 2 may merge into the central lane at the same time. In Figure 2 (b), global consensus is reached that vehicle 2 will merge into the central lane. In this case, lane merging could be performed safely in a collaborative fashion, where vehicles maintain safe distance between them with information of other vehicles' location, velocity and acceleration, i.e., using a car-following model that is similar to CACC [45, 59].

There are more complex scenarios with different degrees of agreement. In Figure 2 (c), partial agreement among vehicles 1, 2 and 4 is reached that vehicle 2 will merge into the central lane. However, as vehicle 3 has the wrong understanding that vehicle 1 will merge into the central lane (or it could be unaware of any



Vehicle	Agreement
1	1 merge
2	2 merge
3	2 merge
4	1 merge

(a)

Vehicle	Agreement
1	2 merge
2	2 merge
3	2 merge
4	2 merge

(b)

Vehicle	Agreement
1	2 merge
2	2 merge
3	1 merge
4	2 merge

(c)

Vehicle	Agreement
1	2 merge
2	2 merge
3	1 merge
4	1 merge

(d)

Figure 2: Lane-merging under different degrees of agreement [29]: (a) Only partial agreement reached, and it is unsafe to perform merging as vehicles 1 and 2 may collide in the central lane. (b) Global consensus/agreement reached, and it is safe to merge based on car-following model similar to CACC. (c) Partial agreement reached among vehicles 1, 2 and 4. Lane merging may be performed based on car-following model similar to ACC between vehicles 2 and 3. (d) Partial agreement reached between vehicles 1 and 2, and between 3 and 4. Lane merging may be performed based on car-following model similar to ACC between vehicles 2 and 3, and between vehicles 2 and 4.

merging at all), vehicle 2 has to be more conservative and assume all behavior of vehicle 3 is possible (i.e., sudden break with maximum breaking force). In this case, instead of CACC, vehicle 2 should adopt a car-following model that is similar to ACC (e.g., the human driver model [24]) to calculate its safe distance with vehicle 3. This will reduce the system performance (e.g., increase the distance between vehicles or the travel time for vehicles), but it should ensure system safety. In Figure 2 (d), there is partial agreement between vehicles 1 and 2 that vehicle 2 will merge into the central lane; while vehicles

3 and 4 assume vehicle 1 will merge. It may still be possible to perform the lane merging with safety guarantee in this case, but vehicle 2 should be most conservative about the possible behavior of vehicles 3 and 4, and assume they could break or accelerate with the maximum force.

We extended the widely-used traffic simulator SUMO [46] and conducted experiments to study how partial consensus may affect the system performance, measured by the average travel time of vehicles. In the simulation, there are 40 vehicles traveling on a 3-lane 500-meter road. The vehicles arrive at the beginning of the road based on a Poisson distribution. The starting lane and the destination lane are randomly generated based on uniform distribution. We assume that under communication disturbance, certain percentage of the lane merging is performed under partial consensus (we only consider the partial consensus case in Figure 2 (c) in this study), while the rest is performed with global consensus. As shown in Figure 3, there is a clear degradation of system performance when the percentage of partial consensus increases.

The example in Figure 2 and the results in Figure 3 demonstrate the impact of communication disturbance on system properties (safety and performance). The varying degree of such disturbance could be captured with different types of partial consensus (note that global consensus and no consensus at all can both be viewed as special cases in partial consensus). However, questions still remain that 1) how to model the communication disturbance *over time*, e.g., how to specify the occurrences of a specific type of partial consensus within a certain time period, and 2) how to analyze system properties given the requirements/assumptions of communication disturbance.

We believe that weakly-hard model could be a promising direction for answering the above questions, as it can provide more deterministic specifications that facilitate formal and quantitative analysis of system properties over time. We could explore leveraging weakly-hard constraints together with the concept of partial consensus to formally analyze and prove the properties of connected vehicle applications. For instance, we may derive weakly-hard bounds for the communication between two vehicles based on the analysis of intra-vehicle task executions and signal transmissions as well as inter-vehicle V2V message transmissions. Such bounds can be used for network-level analysis (such as in Section 2.1) to derive the partial consensus behavior and then reason about system properties. Some of these research directions will be further elaborated in Section 4.

3 RELATED WORK

3.1 Fault Tolerance in Networked Systems

There are a large number of works in the literature that consider the disturbances/faults in networked systems. For instance, some works focus on the tolerance of permanent node failures, where two main methodologies are considered: precautionary and reactive. For the pre-cautionary methods, redundant relay nodes need to be deployed at setup phase for unexpected node failures. The goal is to ensure the connection of the networks with the minimum number of relay nodes [22, 26, 43]. Reactive methods, on the other hand, try to restore network connectivity after node failures occur, by resetting the positions of existing nodes or deploying additional nodes [1, 4, 9, 10]. These approaches differ from one other in the types of constraints (type/number of additional nodes, number of mobile existing nodes, etc.) and the secondary objective other than connectivity (e.g., minimizing average data delivery delay) [51].

Some other works focus on handling transient disturbances. One main direction is to explore new scheduling policies at the architecture layer [11, 40, 52, 53, 62]. In [62], a centralized approach was proposed for dynamically making decisions. However, this approach is time consuming. To improve the efficiency, a partially distributed framework was proposed in [52]. In this framework, a central controller is still employed but only used to compute and propagate necessary high-level information. Each node separately generates its own schedule when receiving the information. This approach effectively distributes the computation load but still suffers from long response time. The authors in [53] then proposed a fully distributed packet scheduling framework to reduce the response time for handling disturbances.

There are also significant efforts directed at designing control algorithms under intermittent errors such as packet losses [16]. The majority of these works model the packet losses as a stochastic process, e.g., Bernoulli process [49, 50, 54] or Markov process [47].

In comparison, we propose to leverage weakly-hard constraints for capturing the bounds on disturbances and formally reasoning about system properties. For systems where such weakly-hard constraints can be derived (e.g., as the example in Section 2.1 or our work in [23]), our approach can provide a more deterministic guarantee on functional properties such as safety, stability, and reachability. Moreover, for cases where a definitive weakly-hard bound is difficult to obtain, we may explore defining weakly-hard constraints in a probabilistic manner (e.g., task τ may miss more than 2 deadlines out of any 10 consecutive activations, with a probability of less than 10^{-9}). This way we may deduce the probability bound for system properties to hold at the network level, based on the probabilities of local weakly-hard constraints being satisfied. Finally, levering weakly-hard constraints also enables “controlled skipping” of operations for improving system adaptability and reducing resource utilization.

3.2 Analysis of Weakly-Hard Models

In the literature, the notion of (m, K) weakly-hard constraints was first introduced in [18]. The authors in [6] formally defined weakly-hard constraints for real-time systems and presented schedulability analysis for periodic tasks under fixed-priority scheduling. Other schedulability analysis works [7, 27, 39, 42] were presented under various assumptions such as bi-modal execution and non-preemptiveness. Weakly-hard constraints were also studied to bound the temporal behavior of overloaded systems [2, 3, 19–21, 35–38, 44, 48], where typical worst-case analysis (TWCA) is conducted for tasks that are activated periodically with sporadic overload. Recently in [?], a job-level scheduling policy was presented for weakly-hard constraints to improve system schedulability.

Besides schedulability analysis, analyzing and optimizing control stability is another topic studied with weakly-hard constraints. The work in [14] analyzes the closed-loop properties of control software based on TWCA. In [39], periodic task instances are statically separated into mandatory and optional instances based on the

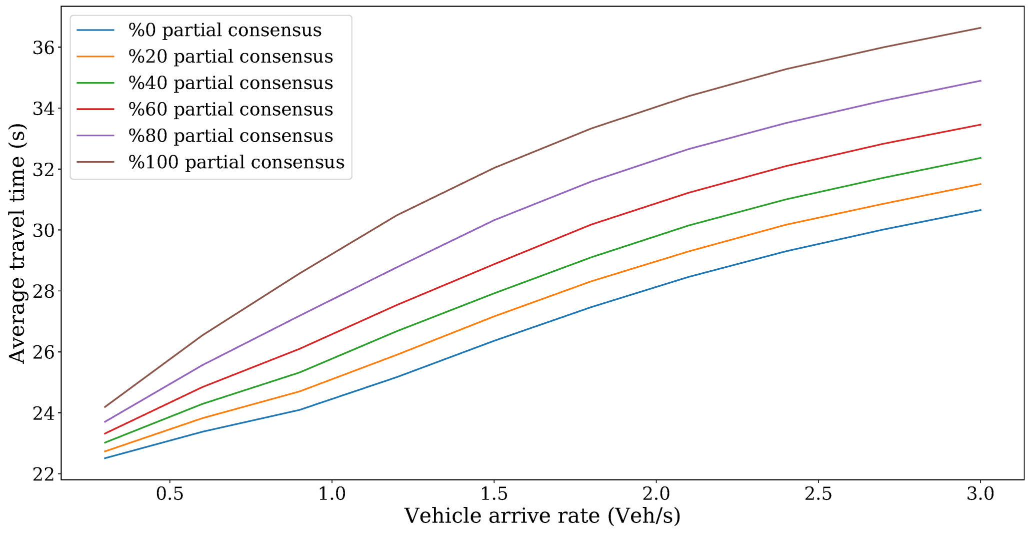


Figure 3: Lane-merging performance under different percentage of partial consensus as specified in Figure 2 (c), while global consensus is assumed to be reached in the rest of cases.

(m, K) constraints, and only the mandatory ones are guaranteed to complete in time. The work in [15] extends this work to improve the performance of optional instances, and the work in [30] considers additional non-periodic execution. In [34], a general state-based weakly-hard model was proposed to measure the performance cost of deadline misses. Several approaches have also been proposed for control-schedule co-design under possible deadline misses [8, 12, 41]. In [8], the authors experimentally studied the trade-off between control performance and resource saving by actively skipping some deadlines. In [12], the relation between deadline misses and sampling periods in terms of control performance was studied. Furthermore, the verification of functional properties such as safety and stability has been considered for linear systems by using satisfiability modulo theories (SMT) solvers [13] and for nonlinear systems by over-approximation based approach [23].

These works on weakly-hard constraints have demonstrated promising results of using them to improve system schedulability and trade off control performance, mostly for single-node systems. We believe that weakly-hard paradigm could also be effective for analyzing and designing networked systems, to provide more deterministic guarantees on system properties and to enable more flexible adaptation for different objectives. In the next section, we will discuss some of the research problems and directions for realizing this vision.

4 RESEARCH PROBLEMS AND DIRECTIONS

4.1 Viewing Networked Systems through the Lens of Weakly-Hard Models

We first describe properties that are relevant in the context of networked systems and then discuss how the weakly-hard model can be used to bound uncertainties and provide deterministic guarantees. Broadly speaking, several classes of properties are important pertaining to the correctness and performance of the overall system. We list some representative ones below.

- Quality-of-service, e.g. bounded latency in routing a data packet from one node in the network to another node in the network;
- Load-balancing, e.g. balancing network traffic across links;
- Availability, e.g. minimum level of uptime;
- Reliability, e.g., tolerance to occasional link failures;
- Consensus such as selecting a leader especially in the presence of faulty processes or faulty links;
- Reachability/isolation, e.g. packets belonging to a certain class, as classified by their header information for instance, should only reach their designated end hosts;
- Self-stabilization, e.g. the networked system would end up in a correct state no matter what its initial state is.

Additionally, the network itself might be utilized to achieve certain control objectives, such as wireless control network [33] or coordination of connected vehicles [17, 57].

As stated before, a long-standing challenge and a subject of active research has been how to design systems that can guarantee these properties even in the presence of uncertainties or faults. Addressing this challenge first and foremost requires the selection of a fault model. Faults can be generally categorized into permanent, transient, or intermittent, depending on their temporal characteristics. It is easy to see that permanent and transient faults (in the non-probabilistic case) can be captured by the (m, K) model. For intermittent faults, the (m, K) model additionally assumes certain regularity of these episodic and recurring faults. We argue that this is a powerful model as it allows the bounding of disturbances without imposing a strong constraint on when faults may actually occur. These bounds are especially important for analysis of safety-critical settings where proving the absence of certain failures (with respect to the high-level properties) is desired. The challenge of computing or verifying these bounds lies in the ability of efficiently over-approximating the effects of the faults as precise as possible, given the potentially large fault space. Our recent work on verifying weakly-hard nonlinear control systems is a step towards this direction [23]. Another orthogonal direction for managing the complexity of verification would be to consider networks with restricted topologies.

In our earlier example, we considered a simple flooding protocol aiming at distributing information quickly to the nodes in an arbitrary network and studied the effects of different (m,K) constraints had on the worst-case latency of the broadcast. We observed that an (m,K) constraint that seemingly allows fewer faults on average actually causes a longer delay on the flooding time. The reason was that the stronger constraint implicitly disallows consecutive faults. Thus, while the (m,K) model is quite expressive, it needs to be refined to models such as $(\hat{m},K)$ to specify at most $\hat{m}$ consecutive deadline misses are allowed in any K consecutive activations [6], and combined with other analyses such as those that bound consecutive message losses [25]. Our earlier work showed that these bounds were critical in assuring end-to-end properties of *multi-rate* systems, i.e. networked systems with bounded latency channels and nodes that execute at different rates [28].

The weakly-hard model provides a new lens through which we can view fault tolerance in networked systems especially when real-time guarantees are of paramount importance. Given the broad range of properties in networked systems, an interesting direction would be to revisit these properties under the weakly-hard model and study the potential trade-offs among different choices of (m,K). It will also be a valuable exercise to compare these analyses with those that rely on probability assumptions (e.g. [32]).

In some sense, redundancy underlies all approaches to fault tolerance. Redundancy typically takes two forms, spatial and temporal. In the weakly-hard paradigm, we envision the development of new fault-tolerant protocols that leverage these redundancies, e.g. those that make use of node replication or message re-transmission appropriately. On the other hand, a system that can preserve its functionality under an (m,K) constraint indicates that there is inherent temporal redundancy in the system. Thus, the *design* question is how do we exploit these redundancies in such a way that we can further optimize other design objectives such as energy consumption (especially for IoT devices) [31], extensibility [61], or security [55]. Answering this question would require tying together analysis at the lower software architecture level for meeting the (m,K) constraints, such as energy-constrained scheduling for weakly-hard systems [5]. We discuss this in detail in the next section.

4.2 Applying Weakly-Hard Paradigm across System Layers

In above, we have argued that a weakly-hard paradigm may facilitate the analysis of system properties under disturbances and the design of more adaptable and efficient networked systems. For both purposes, we believe it is important to take a cross-layer approach that addresses both function and architecture layers.

For analyzing system properties under disturbances, the first step is to derive weakly-hard bounds on operation disturbances, e.g., node failures as in the network flooding example in Section 2.1 or communication failures (long delays or packet losses) as in the connected vehicle applications in Section 2.2. While the analysis for network flooding is conducted at the function layer with weakly-hard assumptions on individual nodes, whether a node may encounter sensing/computation/connectivity faults that lead to deadline misses ultimately depends on implementation details at the architecture layer. As shown in Figure 4, the computation and communication at an individual node α_1 may be further decomposed into a series of software task (thread) execution and internal signal communication. Correspondingly, the analysis of weakly-hard bounds for the failures of this node (e.g., deadline misses or transient faults) can be refined into weakly-hard analysis of software tasks and signals at the architecture layer, i.e., (m,K) for α_1 could be represented as a function f of weakly-hard constraint (m_1,K_1) for the execution of task τ_1, weakly-hard constraint (m_2,K_2) for the signal transmission between tasks τ_2 and τ_3, and weakly-hard (or hard) constraints on other tasks and signals.

Similarly, for the connected vehicle applications, whether two vehicles can exchange information in time depends on the timing and reliability of the architecture layer operations, including the in-vehicle task execution, message transmission over in-vehicle buses, and V2V message communication over vehicular ad-hoc network, as explained in [59]. Thus, deriving the weakly-hard bounds for communication failure at the function layer can also be refined into weakly-hard analysis at the architecture layer. More generally, there could be multiple function layers and architecture layers. Weakly-hard analysis could in principle be applied to each of those layers and cross-layer analysis should be performed to derive the relations between weakly-hard constraints at different layers.

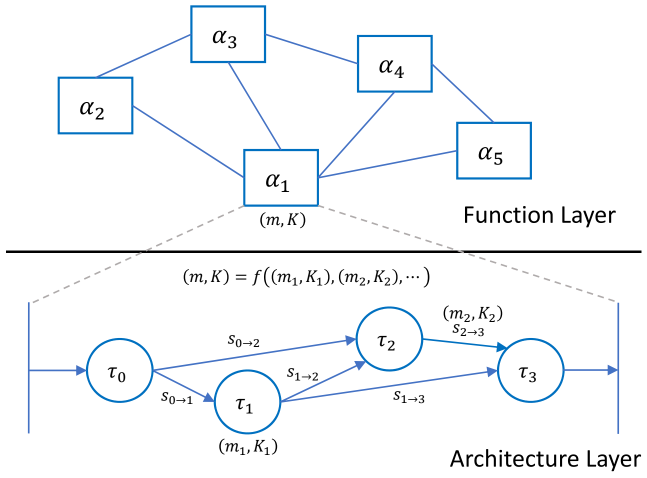


Figure 4: An illustration of how weakly-hard bound of an individual node, e.g., (m,K) constraint for node α_1, at the function layer can be further refined to weakly-hard analysis at the architecture layer, e.g., (m_1,K_1) for the execution of task τ_1 and (m_2,K_2) for the signal transmission between tasks τ_2 and τ_3.

For leveraging weakly-hard constraints to improve system adaptability and reduce resource utilization, it is essential to take a cross-layer approach to ensure that the "controlled skipping" of operations can lead to desired properties at function and architecture layers. For instance, we may intentionally skip some of the node activations/executions during network flooding to reduce energy consumption. It is thus important to ensure that we still achieve the desired flooding performance at the function layer and at the same time maximize the energy savings at the architecture layer. In many cases, we will also need a co-design approach to explore the configurations of both function and architecture layers, as well as the weakly-hard constraints themselves (i.e., choosing how many instances to skip during K activations), to satisfy those properties.

5 CONCLUSION

Networked systems have been playing a significant role in cyber-physical and IoT applications. These systems are often subject to disturbances on their sensing, computation, communication, storage, and actuation operations due to environment interference, security attacks and device faults. Such disturbances may significantly degrade system performance, cause incorrect functional behavior, and even lead to system failure. To address this issue, we present an idea to leverage weakly-hard paradigm for capturing disturbances in a bounded manner and facilitating formal analysis of system properties at the network level. Such approach can also help improve system adaptability and reduce resource utilization by allowing "controlled skipping" based on weakly-hard constraints. We illustrate our idea through two motivating examples in network flooding and connected vehicle applications, and discuss some of the research problems and directions. We believe this is a promising direction for building more predictable and adaptable networked systems.

ACKNOWLEDGEMENT

We gratefully acknowledge the support from the National Science Foundation awards 1834701, 1834324, 1839511, and 1724341. This work is also funded in part by the DARPA BRASS program under agreement number FA8750-16-C-0043 and NSF grant CCF-1646497.

REFERENCES

[1] Ameer A Abbasi, Mohamed Younis, and Kemal Akkaya. 2009. Movement-assisted connectivity restoration in wireless sensor and actor networks. *IEEE Transactions on parallel and distributed systems* 20, 9 (2009), 1366–1379.

[2] Leonie Ahrendts, Sophie Quinton, Thomas Boroske, and Rolf Ernst. 2018. Verifying Weakly-Hard Real-Time Properties of Traffic Streams in Switched Networks. In *30th Euromicro Conference on Real-Time Systems (ECRTS 2018) (Leibniz International Proceedings in Informatics (LIPIcs))*, Sebastian Altmeyer (Ed.), Vol. 106. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 15:1–15:22. https://doi.org/10.4230/LIPIcs.ECRTS.2018.15

[3] L. Ahrendts, S. Quinton, and R. Ernst. 2017. Exploiting Execution Dynamics in Timing Analysis Using Job Sequences. *IEEE Design Test* PP, 99 (2017), 1–1. https://doi.org/10.1109/MDAT.2017.2746638

[4] Kemal Akkaya, Fatih Senel, Aravind Thimmapuram, and Suleyman Uludag. 2010. Distributed recovery from network partitioning in movable sensor/actor networks via controlled mobility. *IEEE Trans. Comput.* 59, 2 (2010), 258–271.

[5] T. A. AlEnawy and H. Aydin. 2005. Energy-constrained scheduling for weakly-hard real-time systems. In *26th IEEE International Real-Time Systems Symposium (RTSS'05)*. 10 pp.–385. https://doi.org/10.1109/RTSS.2005.18

[6] G. Bernat, A. Burns, and A. Liamosi. 2001. Weakly hard real-time systems. *IEEE Trans. Comput.* 50, 4 (Apr 2001), 308–321. https://doi.org/10.1109/12.919277

[7] Guillem Bernat and Ricardo Cayssials. 2001. Guaranteed on-line weakly-hard real-time systems. In *IEEE Real-Time Systems Symposium (RTSS)*.

[8] Tobias Bund and Frank Slomka. 2014. Controller/platform co-design of networked control systems based on density functions. In *ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems*. ACM, 11–14.

[9] Prasenjit Chanak, Indrajit Banerjee, and R Simon Sherratt. 2017. Energy-aware distributed routing algorithm to tolerate network failure in wireless sensor networks. *Ad Hoc Networks* 56 (2017), 158–172.

[10] Xiuzhen Cheng, Ding-Zhu Du, Lusheng Wang, and Baogang Xu. 2008. Relay sensor placement in wireless sensor networks. *Wireless Networks* 14, 3 (2008), 347–355.

[11] Octav Chipara, Chengjie Wu, Chenyang Lu, and William Griswold. 2011. Interference-aware real-time flow scheduling for wireless sensor networks. In *Real-Time Systems (ECRTS), 2011 23rd Euromicro Conference on*. IEEE, 67–77.

[12] Hoon Sung Chwa and Jinkyu Lee Kang G. Shin. 2018. Closing the Gap between Stability and Schedulability: A New Task Model for Cyber-Physical Systems. In *IEEE Real-Time Technology and Applications Symposium (RTAS)*.

[13] Parasara Sridhar Duggirala and Mahesh Viswanathan. 2015. Analyzing real time linear control systems using software verification. In *RTSS*. IEEE, 216–226.

[14] G. Frehse, A. Hamann, S. Quinton, and M. Woehrle. 2014. Formal Analysis of Timing Effects on Closed-Loop Properties of Control Software. In *2014 IEEE Real-Time Systems Symposium*. 53–62. https://doi.org/10.1109/RTSS.2014.28

[15] Mongi Ben Gaid, Daniel Simon, and Olivier Sename. 2008. A Design Methodology for Weakly-Hard Real-Time Control. *IFAC Proceedings Volumes* 41, 2 (2008), 10258 – 10264. https://doi.org/10.3182/20080706-5-KR-1001.01736 17th IFAC World Congress.

[16] Xiaohua Ge, Fuwen Yang, and Qing-Long Han. 2017. Distributed networked control systems: A brief overview. *Information Sciences* 380 (2017), 117–131.

[17] Jacopo Guanetti, Yeojun Kim, and Francesco Borrelli. 2018. Control of connected and automated vehicles: State of the art and future challenges. *Annual Reviews in Control* 45 (2018), 18 – 40. https://doi.org/10.1016/j.arcontrol.2018.04.011

[18] Moncef Hamdaoui and Parameswaran Ramanathan. 1995. A dynamic priority assignment technique for streams with (m, k)-firm deadlines. *IEEE Trans. Comput.* 44, 12 (1995), 1443–1451.

[19] Z. A. H. Hammadeh, R. Ernst, S. Quinton, R. Henia, and L. Rioux. 2017. Bounding deadline misses in weakly-hard real-time systems with task dependencies. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*. 584–589. https://doi.org/10.23919/DATE.2017.7927054

[20] Zain A. H. Hammadeh, Sophie Quinton, and Rolf Ernst. 2014. Extending Typical Worst-case Analysis Using Response-time Dependencies to Bound Deadline Misses. In *Proceedings of the 14th International Conference on Embedded Software (EMSOFT '14)*. ACM, New York, NY, USA, Article 10, 10 pages. https://doi.org/10.1145/2656045.2656059

[21] Zain A. H. Hammadeh, Sophie Quinton, Marco Panunzio, Rafik Henia, Laurent Rioux, and Rolf Ernst. 2017. Budgeting Under-Specified Tasks for Weakly-Hard Real-Time Systems. In *29th Euromicro Conference on Real-Time Systems (ECRTS 2017) (Leibniz International Proceedings in Informatics (LIPIcs))*, Marko Bertogna (Ed.), Vol. 76. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 17:1–17:22. https://doi.org/10.4230/LIPIcs.ECRTS.2017.17

[22] Bin Hao, Han Tang, and Guoliang Xue. 2004. Fault-tolerant relay node placement in wireless sensor networks: formulation and approximation. In *High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on*. IEEE, 246–250.

[23] Chao Huang, Wenchao Li, and Qi Zhu. 2019 (to appear). Formal Verification of Weakly-Hard Systems. In *International Conference on Hybrid Systems: Computation and Control (HSCC)*.

[24] I Ge Jin, Sergei S Avedisov, Chaozhe R He, Wubing B Qin, Mehdi Sadeghpour, and Gábor Orosz. 2018. Experimental validation of connected automated vehicle design among human-driven vehicles. *Transportation research part C: emerging technologies* 91 (2018), 335–352.

[25] R. Larrieu and N. Shankar. 2014. A framework for high-assurance quasi-synchronous systems. In *2014 Twelfth ACM/IEEE Conference on Formal Methods and Models for Codesign (MEMOCODE)*. 72–83. https://doi.org/10.1109/MEMCOD.2014.6961845

[26] Sookyoung Lee, Mohamed Younis, and Meejeong Lee. 2015. Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. *Ad Hoc Networks* 24 (2015), 1–19.

[27] Jian Li, YeQiong Song, and Françoise Simonot-Lion. 2006. Providing Real-Time Applications With Graceful Degradation of QoS and Fault Tolerance According to (m, k)-Firm Model. *IEEE Transactions on Industrial Informatics* 2, 2 (2006), 112–119.

[28] W. Li, L. Gérard, and N. Shankar. 2015. Design and verification of multi-rate distributed systems. In *2015 ACM/IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE)*. 20–29. https://doi.org/10.1109/MEMCOD.2015.7340463

[29] H. Liang, M. Jagielski, B. Zheng, C. Lin, E. Kang, S. Shiraishi, C. Nita-Rotaru, and Q. Zhu. 2018. Network and System Level Security in Connected Vehicle Applications. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.

[30] P. Marti, A. Camacho, M. Velasco, and M. E. M. Ben Gaid. 2010. Runtime Allocation of Optional Control Jobs to a Set of CAN-Based Networked Control Systems.

IEEE Transactions on Industrial Informatics 6, 4 (Nov 2010), 503–520. https://doi.org/10.1109/TII.2010.2072961

[31] B. Martinez, M. MontÃşn, I. Vilajosana, and J. D. Prades. 2015. The Power of Models: Modeling Power Consumption for IoT Devices. *IEEE Sensors Journal* 15, 10 (Oct 2015), 5777–5789. https://doi.org/10.1109/JSEN.2015.2445094

[32] W. Najjar and J. . Gaudiot. 1990. Network resilience: a measure of network fault tolerance. *IEEE Trans. Comput.* 39, 2 (Feb 1990), 174–181. https://doi.org/10.1109/12.45203

[33] M. Pajic, S. Sundaram, G. J. Pappas, and R. Mangharam. 2011. The Wireless Control Network: A New Approach for Control Over Networks. *IEEE Trans. Automat. Control* 56, 10 (Oct 2011), 2305–2318. https://doi.org/10.1109/TAC.2011.2163864

[34] Paolo Pazzaglia, Luigi Pannocchi, Alessandro Biondi, and Marco Di Natale. 2018. Beyond the Weakly Hard Model: Measuring the Performance Cost of Deadline Misses. In *30th Euromicro Conference on Real-Time Systems (ECRTS 2018) (Leibniz International Proceedings in Informatics (LIPIcs))*, Sebastian Altmeyer (Ed.), Vol. 106. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 10:1–10:22. https://doi.org/10.4230/LIPIcs.ECRTS.2018.10

[35] Sophie Quinton, Torsten T. Bone, Julien Hennig, Moritz Neukirchner, Mircea Negrean, and Rolf Ernst. 2014. Typical Worst Case Response-Time Analysis and Its Use in Automotive Network Design. In *Proceedings of the 51st Annual Design Automation Conference (DAC '14)*. ACM, New York, NY, USA, Article 44, 6 pages. https://doi.org/10.1145/2593069.2602977

[36] Sophie Quinton and Rolf Ernst. 2012. *Generalized Weakly-Hard Constraints*. Springer Berlin Heidelberg, Berlin, Heidelberg, 96–110. https://doi.org/10.1007/978-3-642-34032-1_13

[37] Sophie Quinton, Matthias Hanke, and Rolf Ernst. 2012. Formal Analysis of Sporadic Overload in Real-time Systems. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '12)*. EDA Consortium, San Jose, CA, USA, 515–520. http://dl.acm.org/citation.cfm?id=2492708.2492836

[38] Sophie Quinton, Mircea Negrean, and Rolf Ernst. 2013. Formal Analysis of Sporadic Bursts in Real-time Systems. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '13)*. EDA Consortium, San Jose, CA, USA, 767–772. http://dl.acm.org/citation.cfm?id=2485288.2485473

[39] Parameswaran Ramanathan. 1999. Overload management in real-time control applications using (m, k)-firm guarantee. *IEEE Transactions on Parallel and Distributed Systems* 10, 6 (Jun 1999), 549–559. https://doi.org/10.1109/71.774906

[40] Mo Sha, Rahav Dor, Gregory Hackmann, Chenyang Lu, Tae-Suk Kim, and Taerim Park. 2013. Self-Adapting MAC Layer for Wireless Sensor Networks. In *2013 IEEE 34th Real-Time Systems Symposium*. IEEE, 192–201.

[41] Damoon Soudbakhsh, Linh TX Phan, Anuradha M Annaswamy, and Oleg Sokolsky. 2016. Co-design of arbitrated network control systems with overrun strategies. *IEEE Transactions on Control of Network Systems* (2016).

[42] Youcheng Sun and Marco Di Natale. 2017. Weakly Hard Schedulability Analysis for Fixed Priority Scheduling of Periodic Real-Time Tasks. *ACM Transactions on Embedded Computing Systems (TECS)* 16, 5s (2017), 171.

[43] Jian Tang, Bin Hao, and Arunabha Sen. 2006. Relay node placement in large scale wireless sensor networks. *Computer communications* 29, 4 (2006), 490–501.

[44] S. Tobuschat, R. Ernst, A. Hamann, and D. Ziegenbein. 2016. System-level timing feasibility test for cyber-physical automotive systems. In *2016 11th IEEE Symposium on Industrial Embedded Systems (SIES)*. 1–10. https://doi.org/10.1109/SIES.2016.7509419

[45] B. van Arem, C. J. G. van Driel, and R. Visser. 2006. The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics. *IEEE Transactions on Intelligent Transportation Systems* 7, 4 (Dec 2006), 429–436. https://doi.org/10.1109/TITS.2006.884615

[46] Axel Wegener, MichałPiórkowski, Maxim Raya, Horst Hellbrück, Stefan Fischer, and Jean-Pierre Hubaux. 2008. TraCI: An Interface for Coupling Road Traffic and Network Simulators. In *Proceedings of the 11th Communications and Networking Simulation Symposium (CNS '08)*. ACM, New York, NY, USA, 155–163. https://doi.org/10.1145/1400713.1400740

[47] Jing Wu and Tongwen Chen. 2007. Design of networked control systems with packet dropouts. *IEEE Transactions on Automatic control* 52, 7 (2007), 1314–1319.

[48] W. Xu, Z. A. H. Hammadeh, A. KrÃűller, R. Ernst, and S. Quinton. 2015. Improved Deadline Miss Models for Real-Time Systems Using Typical Worst-Case Analysis. In *2015 27th Euromicro Conference on Real-Time Systems*. 247–256. https://doi.org/10.1109/ECRTS.2015.29

[49] Fuwen Yang and Qing-Long Han. 2013. HâĹđ control for networked systems with multiple packet dropouts. *Information Sciences* 252 (2013), 106–117.

[50] Rongni Yang, Peng Shi, Guo-Ping Liu, and Huijun Gao. 2011. Network-based feedback control for systems with mixed delays based on quantization and dropout compensation. *Automatica* 47, 12 (2011), 2805–2809.

[51] Mohamed Younis, Izzet F Senturk, Kemal Akkaya, Sookyoung Lee, and Fatih Senel. 2014. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. *Computer Networks* 58 (2014), 254–283.

[52] Tianyu Zhang, Tao Gong, Chuancai Gu, Huayi Ji, Song Han, Qingxu Deng, and Xiaobo Sharon Hu. 2017. Distributed dynamic packet scheduling for handling disturbances in real-time wireless networks. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE*. IEEE, 261–272.

[53] Tianyu Zhang, Tao Gong, Zelin Yun, Song Han, Qingxu Deng, and Xiaobo Sharon Hu. 2018. FD-PaS: A fully distributed packet scheduling framework for handling disturbances in real-time wireless networks. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 1–12.

[54] Ya Zhang and Yu-Ping Tian. 2010. Consensus of data-sampled multi-agent systems with random communication delay and packet loss. *IEEE Trans. Automat. Control* 55, 4 (2010), 939–943.

[55] Bowen Zheng, W. Li, P. Deng, L. Gérard, Q. Zhu, and N. Shankar. 2015. Design and verification for transportation system security. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1145/2744769.2747920

[56] B. Zheng, C. Lin, S. Shiraishi, and Q. Zhu. 2019. Design and Analysis of Delay-Tolerant Intelligent Intersection Management. *ACM Transaction on Cyber-Physical Systems* (2019).

[57] B. Zheng, C. W. Lin, H. Liang, S. Shiraishi, W. Li, and Q. Zhu. 2017. Delay-Aware Design, Analysis and Verification of Intelligent Intersection Management. In *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. 1–8. https://doi.org/10.1109/SMARTCOMP.2017.7946999

[58] B. Zheng, C. W. Lin, H. Liang, S. Shiraishi, W. Li, and Q. Zhu. 2017. Delay-Aware Design, Analysis and Verification of Intelligent Intersection Management. In *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. 1–8. https://doi.org/10.1109/SMARTCOMP.2017.7946999

[59] Bowen Zheng, Chung-Wei Lin, Huafeng Yu, Hengyi Liang, and Qi Zhu. November 2016. CONVINCE: A Cross-Layer Modeling, Exploration and Validation Framework for Next-generation Connected Vehicles. In *Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on*. Article 37, 8 pages. https://doi.org/10.1145/2966986.2980078

[60] B. Zheng, M. O. Sayin, C. W. Lin, S. Shiraishi, and Q. Zhu. 2017. Timing and security analysis of VANET-based intelligent transportation systems: (Invited paper). In *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 984–991. https://doi.org/10.1109/ICCAD.2017.8203888

[61] Qi Zhu, Hengyi Liang, Licong Zhang, Debayan Roy, Wenchao Li, and Samarjit Chakraborty. 2017. Extensibility-Driven Automotive In-Vehicle Architecture Design: Invited. In *Proceedings of the 54th Annual Design Automation Conference (DAC) (DAC '17)*. ACM, New York, NY, USA, Article 13, 6 pages. https://doi.org/10.1145/3061639.3072956

[62] Marco Zimmerling, Luca Mottola, Pratyush Kumar, Federico Ferrari, and Lothar Thiele. 2017. Adaptive real-time communication for wireless cyber-physical systems. *ACM Transactions on Cyber-Physical Systems* 1, 2 (2017), 8.