Research Statement

Linh Thi Xuan Phan

http://www.cis.upenn.edu/~linhphan/

My research develops theoretical foundations and practical tools for cyber-physical systems (CPS). These systems play an important role in many aspects of our daily lives; examples include medical devices, airplanes, cars, factory automation systems, and intelligent buildings. These systems interact with the physical world and often perform life-critical functions, such as inflating an airbag in the event of a collision. When they fail, or even just respond too slowly, the consequences can be catastrophic. Hence, the goal of my work is to improve the safety, reliability, and security of cyber-physical systems.

Today’s cyber-physical systems are amazingly complex artifacts: for instance, a modern car contains more than 100 microprocessors that run thousands of software functions. As new features are being added and new technologies (such as multi-core processors) are being adopted, this complexity is only going to increase. Because of this trend, existing techniques for designing, analyzing, and verifying CPS – many of which were developed when systems still had a few dozen functions and perhaps one or two processors – are slowly reaching their limits. Thus, designing and analyzing CPS is becoming more expensive and time-consuming, and the number of safety flaws that are making it into deployed systems is increasing. This is evident, for instance, in the enormous number of cars and medical devices that have to be recalled every year.

Research focus. In my work, I develop theoretical methods, analysis tools, and practical design techniques that can be used to build large and complex CPS that are provably safe and secure. There are four ways in which size and complexity can create additional challenges for CPS design: (1) the system can simply have too many components, so existing analysis techniques are no longer practical; (2) the components can change their behavior at runtime, perhaps by running different tasks in response to changes in the environment; (3) new hardware features, such as shared caches or virtualization, can introduce new behaviors that the theoretical models do not anticipate; and (4) the number of ways in which the system can be compromised by an attacker increases dramatically. To address these challenges, I have organized my work into four main research thrusts. I am developing compositional theories and platforms to build scalable analysis methods that can handle large CPS; I am introducing multi-mode techniques that can guarantee safety even when the system changes its behavior; I am developing co-design methods for making system development efficient, cost-effective, and correct-by-construction; and I am working on secure and reliable real-time infrastructures that can support mission-critical and life-critical workloads.

Although much of my work has been focusing on building theoretical foundations, I also like to build concrete systems and work on practical applications; for instance, I have developed a new scheduling platform that applies my work on compositional theories and that is now part of the widely-used Xen hypervisor. I have also been collaborating with researchers from other disciplines, such as formal methods, control systems, databases, and distributed systems.

Impact. I consider real-time and cyber-physical systems to be my core area, and many of my results have been published at the top conferences in this area (RTSS [4, 17–19, 31], RTAS [2, 7, 8, 14, 16, 21, 23, 26], ECRTS [22, 27], ICCPS [29]). However, since my work is interdisciplinary, I have also been publishing at conferences in embedded systems (EMSOFT [25, 30], CASES [12]), design automation (DATE [10]), operating systems (OSDI [5]), and networking (INFOCOM [15]). My work has been funded by NSF, ARO, and ONR, and it has attracted strong interest from industry: for instance, I have ongoing collaborations with General Motors and Toyota, and I have recently started a joint project with Intel on CPS security.

1 Scalable design and analysis of cyber-physical systems

The high complexity of today’s CPS is due not only to additional functionality but also due to two recent trends: (1) functions that used to run on separate processors are now deployed on a common platform, and
different subsystems are increasingly interconnected. These trends have many potential benefits, such as higher efficiency and lower cost, but they also create a complex web of interconnected functions that is too large to be analyzed with existing techniques. Thus, it is difficult to be sure that the CPS as a whole will not fail and will not exceed the necessary response times. The goal of my first research thrust is to develop scalable counterparts to these techniques that can be applied to very large CPS.

**Compositional theory.** One of my key contributions is a new *compositional* approach to scheduling and analysis. In this approach, a complex CPS is broken down into individual components, which are analyzed individually at first. Then, an *interface* is generated for each component that hides the component’s internal details and captures only a few essential properties, such as timing and resource requirements. If the interfaces are chosen carefully, it is then possible to combine them into larger interfaces that cover bigger and bigger subsystems, so that it eventually becomes possible to derive the properties of the system as a whole. This approach can be applied both horizontally (for distributed systems) and vertically (for systems that are scheduled hierarchically), and I have developed compositional theories for both.

I was the first to introduce compositional timing analysis for systems whose components are modeled by very different formalisms [19]. I have pioneered techniques for composing state-based and stateless subsystems – described by Event Count Automata [4] and arrival/service curves in Real-Time Calculus (RTC) [3], respectively – into distributed heterogeneous systems, and this work has sparked a new line of research that is exploring the connection between formal methods (such as timed automata or Lustre programs) and RTC. I also showed in [2] that, using a feedback control mechanism, certain state constraints – such as back-pressure effects caused by blocking writes on finite buffers – can be analyzed using a stateless model, such as RTC. This surprising result is attractive because the analysis is compositional and can be done very efficiently, but it still provides high accuracy. With collaborators at TU Munich, Berkeley, and NUS, I have developed several generalizations of this approach, and I have applied them to a broad range of applications, including automotive [25], avionics and control systems [16], and multimedia systems [9]. My generalized results for automotive architectures have been nominated for the best paper award at EMSOFT 2010 [25].

For hierarchical systems, I have developed a range of novel timing abstractions and resource-aware interface composition techniques [7, 13, 14, 26] that generalize and enhance current work on multiple fronts, including optimality (which minimizes resources), associativity (which enables incremental design), and efficiency. I have also extended these results to compositional scheduling of mixed-criticality systems [21, 27], which provides a foundation for certifying the safety of CPS.

My approach is unique in that it can not only handle heterogeneity, but rather turns it into an opportunity! With the growing complexity of CPS, there is likely no “one size fits all” model that works well for all application domains. Instead, my approach can integrate models and analysis techniques from different disciplines and different domains, and thus leverage the strengths of each model in the scenarios for which it is best suited. For instance, I have demonstrated that, using a combination of formal verification and min-plus/max-plus algebra for the interface analysis [18], it is possible to efficiently analyze systems with far more complex timing behaviors than that was possible before. Similarly, I have shown that a combination of real-time scheduling and traffic shaping (from computer networks) can substantially improve schedulability and efficiency [21].

**Incorporating practical issues into the theory.** Prior work on compositional analysis was purely theoretical: it assumed an idealized platform in which all overheads are negligible. In practice, however, there are many sources of nontrivial overheads – such as preemptions, cache effects, context switches, and interrupts – that can substantially interfere with the execution of tasks. Ignoring these overheads when analyzing a CPS is dangerous because it can lead to wrong results: the analysis may suggest that the system will always meet its timing requirements, but then, at runtime, the system can nevertheless miss important deadlines [26].

To bridge the gap between theory and practice, I developed a novel overhead-aware compositional theory [26] that incorporates the potential platform overheads into the analysis and the interfaces, and thus makes the analysis a lot safer. Another interesting outcome of this work was the insight that, in compositional systems,
certain types of overhead can accumulate in ways that cannot be accounted for by existing methods, such as WCET inflation. I then developed a taxonomy of overheads and novel ways to account for them [26]. I extended the resulting theory to multi-core virtualization platforms [31, 33]; the results were nominated for the best paper award at RTSS 2013.

**Platform and tool support.** To bring the benefits of compositional theory to users and practitioners, I have led the development of the open-source CARTS tool set [24], which implements all the new techniques we have developed. Together with students and colleagues at Penn and Washington University, I have built the RT-Xen compositional scheduling platform [1, 14, 30], which provides a foundation for real-time virtualization on modern hardware. RT-Xen has attracted attention from industry; for instance, we are working with colleagues from General Motors to extend and integrate it into automotive systems. Also, the RTDS real-time virtual machine scheduler from RT-Xen is now part of the official Xen release.

2 Safe run-time adaptation and response via multi-mode theories

Because CPS interact with the physical world (including humans), they need to promptly respond and adapt to changes in their environment. For example, the avionics system in an unmanned aircraft should be able to adapt its configuration and/or its behavior during sudden turbulence, or when one of its processors fails. Adaptation may involve moving or terminating existing tasks, as well as starting new ones, and this must be done without compromising the safety of the system.

**Multi-mode theories.** To formally model and analyze the timing behavior and performance of adaptive systems, I have developed a multi-mode real-time calculus (MMRTC) [18]. By combining the expressiveness of automata with the algebraic and compositional features of RTC, MMRTC enables much more general systems to be analyzed than before, including, e.g., systems with complex bursty event streams and with both resource- and application-level mode changes. I have also generalized these results to settings with complex processing and scheduling semantics [17].

Another important aspect of multi-mode theories is mode change protocols, which describe the steps the system uses to switch from one mode to another. In prior work, these protocols were specified informally, and it was not clear which protocol was safe or best suited for a particular system. To address this problem, I developed a unifying formal semantic framework (MCP) [23] that can be used to formally describe, analyze, and evaluate different mode change protocols. MCP can also be used to design and evaluate new protocols, and to derive optimal protocols for different classes of systems. In my ongoing work, I am building a general platform based on MCP that can be used to efficiently construct, implement, and evaluate run-time adaptation protocols on hardware.

**Generalization to compositional systems.** To enable compositional analysis of adaptive systems, I developed multi-modal real-time interfaces and composition techniques [20, 22]. The connection between multi-mode and compositional reasoning that I introduced in this work stimulated further research on dynamic resource allocation techniques for component-based systems that rely on multi-mode analysis. Together with colleagues at Penn and GM, I am currently working on an application of multi-mode interface theories to automotive systems.

**Generalization to multicore and mixed-criticality systems.** In a modern CPS, tasks can have different levels of criticality; during a resource shortage, priority can thus be given to the more (safety-)critical tasks, while the less critical ones are suspended or run at reduced capacity. Together with a colleague at CMU, I have extended the multi-mode approach to mixed-criticality systems on multicore platforms [8]. This work was the first to support changing the criticality levels at runtime, which is useful in practice but makes the analysis a lot harder. We also developed the first multi-mode mixed-criticality scheduling algorithm, as well as an implementation on multicore hardware.
3 Co-design methods for CPS

Today, it is common to separately design the control software of a CPS and the hardware platform on which it is to be deployed. This has two undesirable consequences: (1) the software and the platform must make conservative assumptions about each other, making the overall design more expensive than it needs to be, and (2) the software is implemented for (and verified on) an abstract platform model that ignores the more intricate details of the actual platform. Thus, properties that have been proven and verified can still be violated by the deployed system.

Control and platform co-design. I am working on efficient co-design methods that tightly integrate the development of control algorithms and platforms. The key insight is that, by making the controller aware of the properties of the platform, it is possible to simultaneously get better control properties and lower resource requirements. With collaborators from MIT and Toyota, I have developed adaptive control algorithms that can accommodate dropped signals, as well as novel platform mechanisms that efficiently implement various skip and abort strategies [28, 29]. We have applied our co-design methods to automotive CPS, and we have shown that they enable a much larger feasible design space while simultaneously reducing design cost by an order of magnitude, compared to existing methods.

Platform-aware model-based development. Model-based development is a way to build systems that are correct by construction. The software is built for a platform-independent model, which is then verified and, as a final step, translated to code for the target platform. However, this approach suffers from the weakness described above: the model typically abstracts away some details of the platform (such as I/O latencies and scheduling delays), and, as a result, the generated code may not actually have the properties that the verification appears to have proven. Together with students and colleagues at Penn, I am developing a “platform-aware” approach to model-based development that takes the interactions between software, platform, and environment into account when generating and verifying code [10–12]. We applied our initial results to medical CPS, such as infusion-pump systems, and we showed considerable improvements in safety. In our ongoing work, we are extending the approach to more complex settings, such as concurrency and distributed platforms.

4 Secure and reliable real-time infrastructures for CPS

Traditionally, CPS have been implemented on dedicated hardware, but it is becoming increasingly common to move some functionality to a shared infrastructure, such as a data center or an external cloud. This trend towards virtualization has many potential benefits for CPS – such as consolidation, elasticity, and better scalability – but current cloud infrastructures cannot provide predictable timing, and it makes it more difficult to defend the CPS against attacks. To address these challenges, I have been working on theoretical foundations and practical techniques for a new cloud infrastructure that can serve as a secure and reliable platform for CPS.

Scalable resource allocation for clouds: Existing real-time resource allocation techniques are not practical for large, complex workloads, and they do not account for virtualization. Hence, my first step was to develop new allocation methods that can scale naturally using cloud/CPS interfaces. The key insight is that both the cloud hardware and the CPS software have a hierarchical structure. Hence, it is possible to use a compositional approach that is somewhat analogous to the approach from the first thrust: we can group both CPS components and cloud resources into larger and larger aggregates and describe their total resource supply or demand via interfaces. At each level of the resulting hierarchy, we can then use the interfaces to perform resource allocation, and we can incrementally refine the interfaces as resources are requested or become available. With students and colleagues at Penn, I have developed a resource allocation technique [15] based on this approach; a case study in the context of network function virtualization showed that this approach has a very small overhead, but it nevertheless provides performance guarantees and even supports dynamic reallocation of resources at runtime. These initial results have been accepted to appear at IEEE INFOCOM’16 [15].
Reliable run-time systems with predictable timing: Real-time resource allocation also places some requirements on the platform itself: it must have a way to bound overheads (e.g., due to cache effects and I/O), and it must be able to provide timing guarantees not only on computation but also on network traffic. I have started to develop a run-time system that has these properties, based on a combination of RT-Xen and software-defined networking. I have developed a time-aware transport protocol that implements path selection and performs dynamic rate reservation based on flow sizes, flow deadlines, and dynamic RTT estimates. To minimize interference from shared caches, my student and I have designed real-time cache allocation strategies [32] based on a combination of page coloring and hardware mechanisms. We have evaluated our approach using concrete implementations for ARM and Intel processors; our results show that these strategies can completely remove cache interference between concurrent applications, which makes the timing substantially more predictable.

Recovering from attacks within bounded time. The safety of many CPS critically relies on timing guarantees. This introduces a new, CPS-specific security risk: an attacker can try to disrupt the system not only by changing its behavior, but also by changing its timing. This risk has received relatively little attention from the security community so far; existing techniques rarely consider timing at all. Together with colleagues at Penn, I have started to develop techniques for detecting attacks on CPS (including timing-related attacks), along with ways to guarantee that the CPS can recover from a successful attack within a bounded amount of time [6]. Our first result was a technique called time-deterministic replay [5]. This technique turned out to have an interesting additional benefit: it can detect a wide variety of covert timing channels, which is a big step towards solving a long-standing problem in the security community. Our ongoing work on defenses and recovery methods is sponsored jointly by NSF and Intel.

References


