A Formal Methodology for Compositional Cross-Layer Optimization*

Minyoung Kim[1], Mark-Oliver Stehr[1], Carolyn Talcott[1]
Nikil Dutt[2], and Nalini Venkatasubramanian[2]

[1] SRI International, USA
{mkim,stehr,clt}@csl.sri.com
[2] University of California, Irvine, USA
{dutt,nalini}@ics.uci.edu

Abstract. The xTune framework employs iterative tuning using lightweight formal verification at runtime with feedback for dynamic adaptation of mobile real-time embedded systems. To enable trade-off analysis across multiple layers of abstraction and predict the possible property violations as the system evolves dynamically over time, an executable formal specification is developed for each layer of the system under consideration. The formal specification is then analyzed using statistical analysis, to determine the impact of various policies for achieving a variety of end-to-end properties in a quantifiable manner. The integration of formal analysis with dynamic behavior from system execution results in a feedback loop that enables model refinement and further optimization of policies and parameters. Finally, we propose a composition method for coordinated interaction of optimizers at different abstraction layers. The core idea of our approach is that each participating optimizer can restrict its own parameters and exchange refined parameters with its associated layers. We also introduce sample application domains for future research directions.

1 Vision

An overarching characteristic of next-generation mobile applications is that they are often data intensive and rich in multimedia content with images, video, and audio data that is fused together from disparate distributed information sources. The content-rich data is expected to be obtained from, delivered to, and processed on resource-constrained devices (sensors, PDAs, cellular handsets) carried by users in highly dynamic environments (e.g., delay, jitter, erroneous transmission). Clearly, in such a scenario, the dual goals of ensuring adequate application QoS (Quality of Service) and optimizing resource utilization in the network, devices, and content servers present significant challenges.

* Support from National Science Foundation Grant 0932397 (A Logical Framework for Self-Optimizing Networked Cyber-Physical Systems) is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

G. Agha, O. Danvy, and J. Meseguer (Eds.): Talcott Festschrift, LNCS 7000, pp. 207–222, 2011.
© Springer-Verlag Berlin Heidelberg 2011

Specific adaptations have been developed within each abstraction layer (application, middleware, OS, hardware) to perform the QoS provision on resource limited devices. For example, the OS adaptations typically change the allocation and scheduling in response to application and resource variations [21,4]. We refer to these individual adaptation techniques as *policies*. Next, we identify *parameters* to manipulate the behavior of a policy. For example, the OS layer policy can be fine tuned by selecting the appropriate tolerance level of QoS in terms of task completion that satisfies its deadline [4].

Understanding interactions across layers and exploiting them in such systems is essential since policy/parameter settings at one layer can have a significant impact on the behavior at other layers. A cross-layer approach is needed to deal with complexity of such systems and the dynamic environment in which such applications execute. Our prior experience (FORGE [3,13]) with developing algorithms for cross-layer adaptation based on QoS/energy trade-offs in distributed mobile multimedia applications has given us valuable insights into the issues to be addressed. In particular, the middleware framework DYNAMO [14] performs joint adaptation at the proxy server to drive on-device adaptation for end-to-end adaptations such as dynamic video transcoding and traffic shaping. GRACE [22] also aims to trade off multimedia quality against energy by introducing a hierarchy of global (i.e., coordinating all layers) and internal (i.e., within the individual layers) adaptation.

While existing work has shown the effectiveness of cross-layer adaptation, many of these efforts try to address the average case behavior without verifiable guarantees on their solutions. As the system evolves dynamically over time, the applications need a mechanism that can be used to formally prove various properties pertaining to energy usage, delays, and so on for any given configuration of policies/parameters to derive, analyze, and validate cross-layer adaptation. Our hypothesis is that a comprehensive design methodology based on a formal reasoning framework will provide an effective basis for tuning mobile embedded systems under a multitude of constraints.

To illustrate the challenges introduced by the cross-layer nature of mobile real-time embedded applications, consider the scenario of a mobile device executing a video conferencing application carried by a user moving from $Zone_0$ to $Zone_4$ in Figure 1. The objective is to support QoS needs by instantiation and tuning of the appropriate policy at each layer with its parameter values (*complexity*). In particular, we strive to achieve quantifiable guarantees with regard to the quality of selected policies and parameters (*verification*). Last, when a user moves to a different zone, we need a way of reflecting it for iterative tuning as well as static instantiation of policies and parameters (*dynamicity*). We elaborate these challenges below.

- **Complexity:** Given a set of application needs and a system configuration, we need to choose appropriate operating points, through selection of both the policy and the parameter settings at each layer as depicted in Figure 1. Considering the composite effect of multiple policies at each layer demands a cross-layer approach. A holistic approach to understanding cross-layer

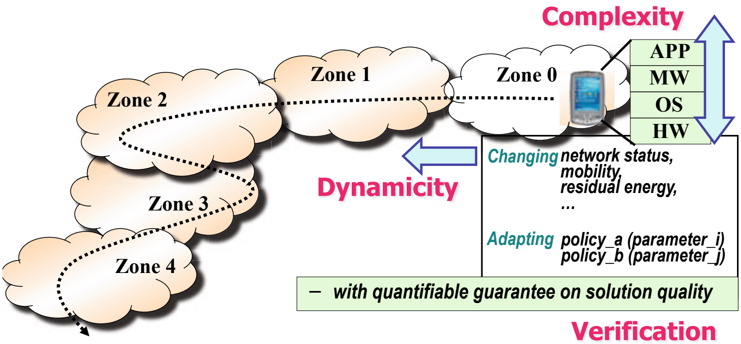
Complexity
Zone 2
Zone 1
Zone 0
APP
MW
OS
HW
Zone 3
Zone 4
Dynamicity
Changing network status, mobility, residual energy, ...
Adapting policy_a (parameter_i) policy_b (parameter_j)
– with quantifiable guarantee on solution quality
Verification

Fig. 1. Challenges: Instantiation and tuning of the appropriate polices and corresponding parameter values (*complexity*) with quantifiable guarantee (*verification*) while reflecting changes in the system and environment (*dynamicity*)

interaction in such systems is essential, since policies made at one layer can (sometimes adversely) affect behavior at other layers.

- **Verification:** During this process, we need to generate a set of candidate policies with possible parameter settings based on the trade-off analysis to determine the best feasible choice among these candidates. If no policy can satisfy the requirements, we must determine how to relax the constraints and may need to repeat policy selection. For such informed selection, it is essential to perform bound/sensitivity analysis on the impact of the policy/parameter that can provide some notion of guarantee on the solution.
- **Dynamicity:** The system and environment may keep evolving as a user moves from one zone to another as depicted in Figure 1, requiring dynamic policy/parameter analysis and tuning. During operation, policy selection and parameter tuning requires the procedure to determine (i) which changes demand our attention, (ii) if it has a significant impact on timing/QoS/performance, and (iii) how the policy/parameter should be recomputed.

To ensure adequate application QoS and resource utilization with timing and reliability concerns, the ability to compensate *on the fly* for property violations at different layers of abstraction is of paramount importance since there are several sources of unpredictability (e.g., delay, packet drop, user mobility) in a mobile embedded system that introduce nondeterminism. Furthermore, system-level optimizations for effective utilization of distributed resources can interfere with the properties of executing applications. For instance, dynamic voltage scaling (DVS) mechanisms slow down processors to achieve power savings, but at the cost of increased execution times for tasks. Many applications have flexible QoS needs that dictate how tolerant they are to delays and errors — the lack of stringent timing needs can be adaptively exploited for better end-to-end resource utilization.

We enumerate sample questions we would like to answer:

1. How does one decide what policies and parameters to assign to each abstraction layer to minimize the overall energy consumption while providing a sufficient level of QoS with verifiable/quantifiable solution quality? This must be achieved for an energy-constrained mobile embedded device dealing with displaying delay-sensitive multimedia data over a lossy network.
2. How can we exploit system state for dynamic adaptations? When the system evolves over time, how can we accommodate it? We need a way of reflecting dynamics. Specifically, this requires determining which attribute can be a trigger for adaptations and how to refine our model.
3. How can we support cross-layer adaptation while individual policies perform their own optimization? Unlike existing research literature that relies on a global coordinator at a certain layer, we address the issue of how to support cross-layer adaptation while allowing autonomy of individual layers' policies.

To lend focus, we (i) choose mobile multimedia as an application domain, and (ii) select performance criteria that require adaptations such as device residual energy, application QoS/timing needs, and reliable content delivery. A preliminary study [5] demonstrated the need for integration of formal methods with experimentally based cross-layer optimization techniques [3,13] for such application domain and performance criterion. Within the xTune framework, we support compositional online optimization of individual policies at each layer [11]. xTune employs statistical formal methods to analyze given cross-layered optimization policies with a quantifiable guarantee on the solution quality [10].

2 Overview of Technical Approach

Our approach starts with a formal specification of the abstraction layers and subsequent statistical evaluation to verify probabilistic properties. We propose a lightweight formal methodology in the sense that we exploit statistical techniques on a system model represented in an executable formal specification. The proposed approach provides statistically meaningful answers from on-demand trace generation rather than keeping the entire spectrum of possible traces.

Our approach supports iterative tuning and compositional cross-layer optimization and can deal with quantifiable guarantees, dynamicity, and complexity issues:

- **Quantifiable Solution Quality:** Our work examines the impact of various resource management techniques on end-to-end timing/QoS properties based on statistical evaluation for verifiable/quantifiable solutions, and enables informed selection of resource management policies along with rules for instantiation of parameters that derive the policies.
- **Iterative Tuning:** We enhance such lightweight formal modeling and analysis by integrating it with observations of system execution behavior to achieve adaptive reasoning by providing more precise information on current execution and future state.

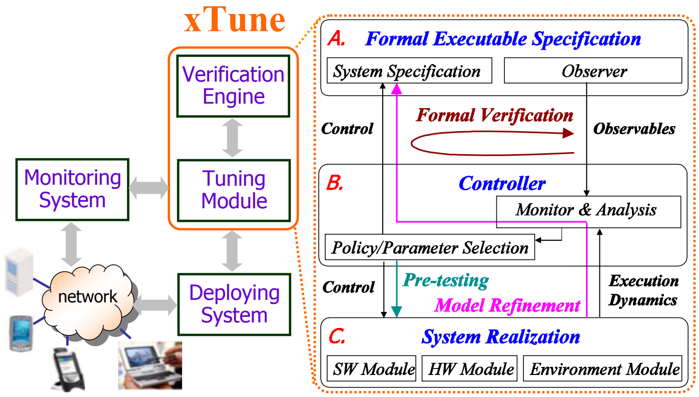

Fig. 2. xTune Cross-layer System Tuning Framework

- **Compositional Cross-Layer Optimization:** We propose a compositional approach for the cross-layer optimization that avoids high overhead introduced by traditional global approaches; our compositional approach allows sublayers' optimization results to be used by the other sublayer optimizers as constraints.

Our work is validated and tested in the context of distributed mobile multimedia applications that have wide consumer interest. Using multimedia as a demonstrator, we have developed general principles and a framework. In many contexts, enabling verifiable adaptation in terms of timing/QoS guarantees provides an additional degree of confidence to improve other cross-layer reliability measures in the context of multimedia applications.

3 Supporting Model-Based Composition with xTune

The xTune framework uses iterative system tuning to support adaptations. In particular, our approach tunes the parameters in a compositional manner allowing coordinated interaction among sublayer optimizers. The xTune framework initially performs property checking and quantitative analysis of candidate policy/parameter settings via formal executable specifications followed by statistical techniques. Iterative tuning allows model refinement from up-to-date and continuous observations of system execution behavior. Furthermore, the results can be used to improve adaptation by verifying given system properties or by relaxing constraints.

Figure 2 presents the overall flow of our approach. *Box A* represents the formal modeling. The core of our formal modeling approach is to develop formal executable models of system components at each layer of interest. These models express functionality, timing, and other resource considerations at the appropriate level of detail and using appropriate interaction mechanisms (clock ticks, synchronous or asynchronous messages). Models of different layers are analyzed in isolation and composed to form cross-layer specifications. We use the Maude system for developing and analyzing formal specifications. One advantage of formal executable models is that they can be subjected to a wide range of formal analysis, including single execution scenarios, search for executions leading to states of interest, and model checking to understand properties of execution paths.

Box B in Figure 2 shows the evaluation phase of given specifications to generate statistics for properties and values of interest. Specifically, we have developed new analysis techniques (statistical model checking and statistical quantitative analysis) that combine statistical and formal methods, and applied them to a case study of a videophone application [10]. We have developed a compositional cross-layer optimization by coordinated interaction among local (sublayer) optimizers through constraint refinement. The constraint refinement allows encapsulation of detailed system state information. In compositional optimization, each local optimizer uses refinement results of other optimizers as its constraints. The constraint representation can be used as the generic interface among different local optimizers, leading to substantial improvement of solution quality at low complexity.

Using such models and analysis, tools can be developed to achieve adaptive refinement of an end-to-end system specification into appropriate policy/parameter settings. We use an *iterative* tuning strategy that combines formal methods (verification) with dynamic system execution behavior (obtained by either simulation or implementation). The execution behavior from system realization (*Box C* in Figure 2) is fed back into the formal modeling to refine the executable specification (*model refinement*). In addition, we can assure the quality of a new policy/parameter constructed by the controller. In Figure 2, *pre-testing* on a system realization can lead to improvements because typically the formal model cannot cover all the possible implementation details of a real system.

4 Model-Based Compositional Cross-Layer Optimization

4.1 Understanding the Issue of Cross-Layer Optimization

To enhance system utility capturing the effectiveness of the settings relative to the user and system objectives in the context of mobile applications, researchers have proposed a wide variety of techniques at different system layers. Note that one key performance metric for such techniques is how well they manage utility under a multitude of constraints in a dynamic situation. Since utility comes with cost in terms of performance, energy consumption, storage requirements, and bandwidth used, one needs to optimize utility in the context of the operating conditions. However, most optimization techniques consider only a single system layer, remaining unaware of the strategies employed in the other layers. A cross-layer approach that is cognizant of features, limitations, and dynamic changes at each layer enables better optimization than a straightforward composition of individual layers, because solutions for each individual layer can be globally suboptimal. To coordinate the individual techniques in a cross-layer manner based on the operating condition, one needs to

- Quantify the effect of various optimization policies at each layer on system properties
- Explore methods of taking the impact of each policy into account and compensating for it at other layers

Abstraction and Model Refinement. We develop a formal methodology to specify and analyze features/constraints/needs at each layer and to correlate them across layers to realize cross-layer tuning. Our approach is to start with an executable formal model based on rewriting logic specifying a space of possible behaviors. In [10,9,11], we use the Maude [2] rewriting logic formalism to develop executable specifications of each layer in isolation and in composition as well as representing their timed behavior.

In most cases, the model cannot be fully characterized in advance and can change while the system is in operation, which is why model refinement is an essential component of our architecture. To reflect execution dynamics, we perform model refinement from observed system execution behavior by equipping the controller with a feedback loop to experiment with the system realization [9]. The system can start with a default model (e.g., a model with default parameters about execution times), which is incrementally refined during the operation of the system. Models can be passively refined by observations — e.g., from CPU usage, while the system is executing its primary function or mission — or it can be actively pursued by exploration, which may require physical actions. Often, combinations of the passive and active modes of model refinement will be needed for acceptable performance with low exploration overhead. Within our framework, there are at least two roles for feedback from observation of system execution behavior: it can be used to improve the model (to make it more accurately match the real environment) and it can be used to directly improve the policy. We define the former as *long-term tuning*, and the latter as *short-term* tuning. In the xTune framework, we support long-term tuning through model refinement without active exploration.

Statistical Analysis. To analyze the behavior of the system (e.g., in terms of discrete or continuous observable properties) in a probabilistic sense, we have implemented two lightweight formal analysis techniques: statistical model checking and statistical quantitative analysis. To formally verify certain properties, traditional approaches maintain tree-like structures of the entire spectrum of possible traces with probability measures and exhaustively evaluate the system, which leads to excessive memory requirements that limit scalability of the solution. In contrast, the xTune approach is a lightweight formal method, since the optimization problem for adaptation does not require an exact solution. This allows us to generate traces on demand and provide statistically meaningful answers, unlike exhaustive numerical methods, which aim at exact solutions.

In [10], we extended the quantitative approach of [1] by an on-demand sample generation that can compute the sample size sufficient to reach confidence in the normality of data, and then utilize the normal distribution to obtain the error bound and confidence interval for quantitative analysis. The xTune framework also implements two statistical model-checking techniques: the sequential probability ratio test [20] and black-box testing [16]. Given a property, the sequential probability ratio test [20] continues sample generation until its answer about accepting or rejecting the hypothesis can be guaranteed to be correct within the required error bounds. Black-box testing [16] instead computes the statistical significance (p-value) for a given number of samples without having any control over the execution. These statistical techniques can be used to quantify statistical performance (e.g., execution times) with a specific confidence and to verify properties (e.g., battery depletion), which may be satisfied only in a probabilistic sense. They provide a quantifiable solution to enable policy-based operation and adaptation as well as parameter setting and adjustment for selected policies.

4.2 Constraint Refinement and Composition

In the xTune framework, constraint-based optimization is guided by a model of the system to be optimized. The compositional optimization is purely generic in the sense that we can construct an interface language for generic composition (e.g., negotiation and contract), which can be used with heterogeneous application specifications. An interesting extension would be distributed compositional techniques that integrate randomization and symbolic reasoning. For that purpose, the constraint language needs to be expressive enough to support strategies for distributed cooperative optimization.

In the following, we describe our composition method. First, we explain the idea of constraint refinement for robust optimization. Then, we define our compositional cross-layer optimization based on this representation. Our experiments show that the encapsulation of the local optimization at each sublayer leads to substantial improvement of solution quality at low complexity [11].

Constraint Refinement. Given an optimization problem with the model $\mathcal{M}$ and the parameter space $\mathbb{P}$, our approach attempts to quickly find a region $P \in \mathcal{R}(\mathbb{P})$[1] containing a nearly optimal solution by the following heuristics:

1. **Recursive Resampling:** We obtain observables by Monte Carlo sampling over the current region $P_i \in \mathcal{R}(\mathbb{P})$ using the model $\mathcal{M}$. Subsequently, we refine P_i to P_{i+1} such that the utility is maximized based on the samples available, and $size(P_{i+1}) = size(P_i) \cdot \tau_i$, where τ_i $(0.0 < \tau_i < 1.0)$ represents the i-th refinement ratio. The new region P_{i+1} is then used as the current region and the process is repeated.

[1] Region $P \in \mathcal{R}(\mathbb{P}) \iff P \subseteq \mathbb{P}$ is a closed convex set, (i.e., if $(x, z \in P) \bigwedge (x < y < z)$, then $(y \in P)$) and P is finitely representable (e.g., interval-based).

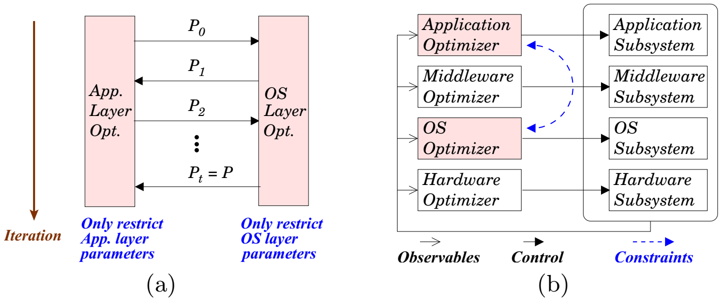

Fig. 3. Constraint Refinement for Composition (a) Parallel Composition of Layers, (b) Compositional Cross-layer Optimization

2. **Interval-based Description:** For simplicity we use regions defined by the Cartesian product of intervals for each of the parameters. For example, an application layer region might be

   $$P_{App} = [Param1_{min}, Param1_{max}] \times [Param2_{min}, Param2_{max}].$$

   More expressive constraint languages are possible in our framework and should be investigated in the future.
3. **Generic Constraint-based Interface:** The input (P_i) and output (P_{i+1}) of each refinement step are regions (infinite sets), and our approach lifts the level of abstraction by treating P_i as *constraints* (finite symbolic representations) when we restrict the resampling space to find P_{i+1}.

The process of constraint refinement can be stated as a chain

$$\mathbb{P} = P_0 \supseteq P_1 \supseteq P_2 \supseteq \cdots \supseteq P_t = P$$

where P is the set of admissible parameter settings at termination after t iterations.

Our experimental results indicate that the constraint refinement can be effectively used for robust parameter selection by refining spectrum of reliable policies and parameters. One key feature of this approach is that we can coordinate parallel composition of individual optimizers as illustrated in Figure 3(a). Each sublayer optimizer controls a subset of parameters. For instance, the application layer optimizer only restricts its own parameters ($\mathbb{P}_{App}$), while the OS layer optimizer only restricts OS-related parameters ($\mathbb{P}_{OS}$). The constraints P_i are used as inputs *and* outputs of individual (sublayer) optimizers.

Composition through constraint refinement reduces the possibility of conflicts because of the more general notion of a solution compared with traditional single-point optimizers. More important, constraint refinement enables simple yet powerful cross-layer optimization via composition (Figure 3(b)), as discussed below.

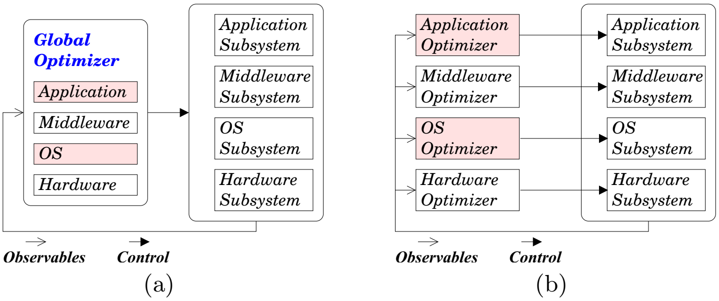

Fig. 4. Comparison among Online Optimizations (a) Global Cross-layer Optimization, (b) Without Cross-layer Optimization

Online Cross-Layer Optimization. The primary goal of our framework is to enable online cross-layer optimization that provides the refined parameter settings from which a system can select any suitable operating point within the region as explained above. The constraint refinement allows encapsulation of detailed system optimization information. This opens up the possibility of co-ordinated interaction (composition) instead of relying on a global view. Figures 4(a), 4(b), and 3(b) compare the global vs. local vs. compositional approach for cross-layer optimization, respectively. The key idea underlying the *compositional* optimization is to exchange the local optimizer's decision for an informed selection. This allows us to achieve a balance between *global* optimization's full awareness with high overhead and *local* optimization's minimal complexity with poor solution quality.

The sampling strategy explores the search space of potential solutions by constraining the behavior of local optimizers in accordance with the other optimizers' refinement results. Thus, the constraint language can serve as a generic interface among different local optimizers, leading to improvements of solution quality and convergence speed. In comparison, a global cross-layer optimizer that resides at a certain layer that is fully aware of the complex system dynamics can introduce unacceptable overhead.

A similar strategy can be applied to other optimization techniques (e.g., simulated annealing [12]). The strict convergence to a single point, however, may not be achievable in the sense that at each step the intermediate parameter settings may be totally different from the previous iteration. These types of abrupt and/or constant parameter changes are not desirable in practice. Constraint refinement can still undergo constant parameter changes, but with *lower* impact since any parameter settings (p_i) within the region (P_i) can be chosen, and the probability that p_i is valid after the next iteration ($p_i \in P_{i+1}$) is proportional to the refinement ratio. We can also easily see that the situation will worsen with conflicting local objectives.

Our approach is not limited to a specific constraint refinement protocol scheme. Compositional optimization through constraint refinement enables a controller to coordinate existing optimizers (possibly distributed) that can accommodate different objectives by treating them as black boxes, which in turn permits them to operate in parallel. Different solutions obtained concurrently can be unified by taking the intersection, which corresponds to the conjunction at the symbolic level.

4.3 Integration of Formal Analysis with System Dynamics

To reflect execution dynamics, the xTune framework performs model refinement from observed system execution behavior by equipping the controller with a feedback loop to experiment with the system realization as highlighted in Figure 2. Within our framework, there are two roles for feedback from observation of system execution behavior: it can be used to improve the model (to make it more accurately match the real environment) as we presented in [9], and it can also be used to directly improve the policy. In [9] the formal specification is refined by replacing timing information with observations from dynamic system execution either by system realization (e.g., simulator) or real implementation, in order to more realistically reflect the actual executions characterizing the system in practice such as data dependent execution times.

5 Sample Application Domains

Here we lay out possible future research directions that we believe can benefit from model-based compositional cross-layer optimization.

5.1 Networked Cyber-Physical Systems

As elaborated in [18], system control and optimization in networked cyber-physical systems (NCPS) is a challenging task. Traditional optimization techniques that strive for optimal solutions based on precise models are not suitable for most real-world problems, where models have many dimensions of uncertainty, and optimality is neither desirable nor achievable. What is needed in practice are strategies to find acceptable and robust solutions that are sufficient to achieve the goal while taking into account the limitations of the models and of the available resources. Capabilities to explore the state space of NCPS have fundamental limitations. The exploration can be expensive in terms of computation, and physical actions can be costly in terms of time, energy, and other resources or even harmful to humans or to the environment.

Furthermore, the overall goal of NCPS cannot be simply decomposed top-down into goals that are optimized locally at each node and each layer, because solutions may require cooperation across layers and across nodes. It is important to keep in mind that even abstract models can be quite complex with multiple and nonoverlapping regions of potential solutions so that purely local gradient-based optimization strategies are clearly insufficient. Given that modifications in parameters (e.g., node position) cannot always be achieved instantaneously, reaching a new improved solution may require transition through intermediate states with lower utility (e.g., lower performance). The distributed nature of NCPS, the limited communication capabilities, the uncertainties in the environment, and the possibility of failures further exacerbate this situation, because system operation is inherently asynchronous.

On the other hand, NCPS with a large number of nodes offers many advantages including fault tolerance, distributed sensing, coordinated actions, and inherent parallelism for computational processes. Technically, a vast range of capabilities is already available at the hardware level, but the challenge to design a software architecture that can exploit those capabilities and to present them as a single cyber-physical system is far from being met. In this regard, [7] provides a prototype of a distributed logical framework based on the partially ordered knowledge-sharing model and an API for cyber-physical devices that enables interaction with the physical world (see `http://ncps.csl.sri.com` for details). The proposed API provides a uniform abstraction for a wide range of NCPS applications, especially those concerned with distributed sensing, optimization, and control. Using the API with or without a distributed logic, NCPS can be programmed to adapt to a wide range of operating points between autonomy and cooperation to overcome limitations in connectivity and resources, as well as uncertainties and failures [7,17,8]. Along this line of research, our methodology can be extended to consider multiple distributed cyber-physical nodes as local optimizers (horizontal composition in addition to vertical (layered) composition). To capture the distributed and heterogeneous nature of NCPS, the compositional optimization strategies need to be generalized to include composition among various local optimizers across layers as well as across nodes, leading us to distributed cooperative constraint refinement.

5.2 Dependable Instrumented Cyber-Physical Spaces

The ability to integrate sensing and communication platforms with large-scale distributed storage/computing facilities and software services enables the creation of instrumented cyber-physical spaces (ICPS). Applications dictate application-specific constraints on the timeliness and accuracy/quality at which information must be captured and delivered from the infrastructure. Repurposing the infrastructure and its software/hardware resources dynamically to realize different application functionalities presents challenges. In this context, it is natural to develop a framework that can customize the operation of ICPS to meet the varying needs of applications and users, based on an observe-analyze-adapt philosophy [6]. The xTune formal modeling and analysis framework can be extended to support specification of properties at both infrastructure and application levels, including multidimensional QoS properties and the relationships among them. Tuning processes at both the application and infrastructure levels can use compositional optimization to derive and validate the tuning and adaptation factors (sensors, policies, parameters).

Among many crosscutting concerns (e.g., security, privacy), let us take an example of cross-layer and end-to-end dependability issues. ICPS should be dependable despite disruptions/failures in sensing, communication, and computation.

Dependability of ICPS thus includes attributes such as availability, reliability, maintainability, safety, and integrity [15]. Realizing dependability requires monitoring and management of parameters at different layers of the system. Composition of nonfunctional needs such as dependability cannot be addressed in a single layer or device due to the inherent dependencies/trade-offs among them (e.g., techniques at any layer to improve dependability usually have implications on timing and power.). At the infrastructure level, a broad array of devices is interconnected by various communication channels (e.g., Ethernet, cellular, Wi-Fi) with distributed middleware support to execute cyber-physical applications. We view each device as a vertically layered architecture consisting of application, middleware, network, OS, and hardware layers. At each layer, the system can enforce policies that are (i) independent of other layers, (ii) a vertical composition of policies on the device across layers, and (iii) a horizontal composition of policies distributed across nodes.

To illustrate dependability across layers, consider the following example. If the data has high importance with a short expiration time, the middleware layer must adjust the frequency of dissemination appropriately. Similarly, CPU slowdown to control thermal runaway (hotspot) at the hardware layer may increase deadline misses in the OS task scheduling layer; this anomaly bubbles up to the application layer and is manifested as a failure to provide up-to-date data. Furthermore, deadline misses may lead to the delayed delivery of the network packets, which in turn results in a failure for timely delivery of messages. From a dependability perspective, both permanent and transient errors need to be modeled and mitigated. For instance, heavy utilization of the device hardware (e.g., for peak performance) can result in excessively high temperatures that may cause thermal errors; to alleviate this, we may trigger task replication or re-execution at the OS layer. The mitigation strategy might cause packet loss due to buffer overflow, since it requires more processing time. Under such circumstances, the dynamic choice of routing algorithms and their parameters needs to consider higher-layer QoS constraints, (partial) knowledge about the network (e.g., sensor density, coverage), heterogeneous devices (with different error sources), and operational context (e.g., prioritizing information flow).

5.3 Physical Infrastructure Protection

Physical infrastructure availability relies on the process control systems that can gather, handle, and share real-time data on critical processes from and to networked entities. For example, wireless sensor networks are now being applied in the industrial automation to lower systems and infrastructure costs, improve process safety, and guarantee regulatory compliance. Harsh environments such as remote areas with potential toxic contamination where mobile ad hoc networks can be the only viable means for communications and information access often necessitate the use of mobile nodes (e.g., surveillance robots with camera and position-changing capability). Optimized control based on continuous observation is an integral part because availability is becoming a fundamental concern to reduce the vulnerabilities of such systems.

Let us take an example of a surveillance system, consisting of a collection of sensors deployed at fixed locations together with mobile nodes, that monitors critical national infrastructure by distributed sensing and actuating. Due to possible jamming attacks and mobility of nodes, the wireless sensors and mobile nodes need to communicate via opportunistic links that enable the sharing and evaluation of data such as video streams in the presence of unstable connectivity. The challenge here is enabling networked entities to respond to dynamic situations in an informed, timely, and collaborative manner so that the physical infrastructure can safely recover after a cyber-disruption. The idea of automated verification and configuration of situation- and resource-aware cross-layer security needs to be investigated since security goals at each layer can be counterproductive and even harmful.

Furthermore, the implementation of security goals is constrained by the available resources. Various solutions ranging from event-driven or on-demand power cycling to reduce transmission power are possible, but the security effects cannot be understood at a single layer. This is why security should be viewed as a multidimensional cross-layer objective for which reasonable trade-offs must be found in a situation- and resource-aware manner. The resources of the wireless sensors and mobile nodes need to be provisioned to ensure a certain level of security while avoiding the depletion of residual energy and avoiding congestion. This requires the dynamic configuration of individual (seemingly independent) techniques to compose the appropriate protection against attack situations while also making optimal use of resources. By supporting specification of security properties across layers and exploiting the composition methods among them, the response to cyber-disruption is adapted to the situation and resource constraints.

6 Concluding Remarks

We have elaborated on the need for a unified framework for analyzing, deriving, and validating cross-layer adaptations for mobile applications operating in highly dynamic environments. Specifically, we have presented the design principles and implementation of the xTune framework [19]. We have developed formal analytical methods for understanding cross-layer optimization issues in mobile real-time embedded systems that incorporate resource-limited devices, and to integrate these methods into the design and adaptation processes for such systems. We have focused on the primary problem of identifying how to tune policies and parameters for cross-layer adaptation that aims to manage resource usage, and to satisfy the multifaceted constraints while providing a sufficient level of QoS with a verifiable/quantifiable solution quality.

We have presented our approach of iterative system tuning for mobile real-time embedded systems that has been applied in a case study treating the videophone mode of a multimode multimedia terminal. The integration of lightweight formal methods with the observation of dynamic system execution results in a feedback loop that includes the formal models, simulation, and monitoring of running systems. Within the xTune framework, we proposed compositional cross-layer optimization to achieve robust and sufficiently good parameter settings with low overhead by coordinated interaction among local optimizers through refinement of constraints that can be used further as a basis of local optimization.

The underlying formal executable models are moderately simple to develop, and their analysis is feasible. The experiments on a fairly complex case study demonstrate the applicability of our framework to cross-layer adaptation of mobile real-time embedded systems. The work on xTune complements our previous work on experimentally based cross-layer strategies (FORGE [3]) and conclusively shows that the xTune framework provides a uniform methodology for deriving, analyzing, and validating cross-layer adaptation.

The xTune framework essentially combines simulation, monitoring, and execution with formal methods. Lightweight formal analysis seems sufficient for multimedia applications in general. However, in the presence of mission-critical applications, context awareness and situation awareness (e.g., live video feed should be undisturbed in case of emergency evacuation) need to be further explored. Even though our current study using the xTune framework has produced encouraging results, the discussions in Section 5 present strong motivation for future work as mentioned in the sample application domains.

References

1. Agha, G., Meseguer, J., Sen, K.: PMaude: Rewrite-based specification language for probabilistic object systems. In: 3rd Workshop on Quantitative Aspects of Programming Languages, QAPL 2005 (2005)
2. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.: All About Maude - A High-Performance Logical Framework. How to Specify, Program and Verify Systems in Rewriting Logic. LNCS, vol. 4350. Springer, Heidelberg (2007)
3. Forge Project, `http://forge.ics.uci.edu`
4. Hua, S., Qu, G., Bhattacharyya, S.S.: Energy reduction techniques for multimedia applications with tolerance to deadline misses. In: Proceedings of the 40th Conference on Design Automation (DAC 2003), pp. 131–136 (2003)
5. Kim, M., Dutt, N., Venkatasubramanian, N.: Policy construction and validation for energy minimization in cross layered systems: A formal method approach. In: IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2006) Work-in-Progress Session (2006)
6. Kim, M., Massaguer, D., Dutt, N., Mehrotra, S., Ren, M.-O.S.S., Talcott, C., Venkatasubramanian, N.: A semantic framework for reconfiguration of instrumented cyber physical spaces. In: Workshop on Event-based Semantics (WEBS 2008), CPS Week (2008)
7. Kim, M., Stehr, M.-O., Kim, J., Ha, S.: An application framework for loosely-coupled networked cyber-physical systems. In: 8th IEEE/IFIP Int. Conf. Embedded and Ubiquitous Computing, EUC 2010 (2010)
8. Kim, M., Stehr, M.-O., Talcott, C.: A distributed logic for networked cyber-physical systems. In: IPM Int. Conf. Fundamentals of Software Engineering, FSEN 2011 (2011)
9. Kim, M.-Y., Stehr, M.-O., Talcott, C., Dutt, N., Venkatasubramanian, N.: Combining Formal Verification with Observed System Execution Behavior to Tune System Parameters. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 257–273. Springer, Heidelberg (2007)

10. Kim, M.-Y., Stehr, M.-O., Talcott, C., Dutt, N., Venkatasubramanian, N.: A Probabilistic Formal Analysis Approach to Cross Layer Optimization in Distributed Embedded Systems. In: Bonsangue, M.M., Johnsen, E.B. (eds.) FMOODS 2007. LNCS, vol. 4468, pp. 285–300. Springer, Heidelberg (2007)
11. Kim, M., Stehr, M.-O., Talcott, C., Dutt, N., Venkatasubramanian, N.: Constraint refinement for online verifiable cross-layer system adaptation. In: Design, Automation and Test in Europe Conference and Exposition, DATE 2008 (2008)
12. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science (4598), 671–680 (1983)
13. Mohapatra, S., Cornea, R., Oh, H., Lee, K., Kim, M., Dutt, N.D., Gupta, R., Nicolau, A., Shukla, S.K., Venkatasubramanian, N.: A cross-layer approach for power-performance optimization in distributed mobile systems. In: IEEE 19th International Parallel and Distributed Processing Symposium, IPDPS 2005 (2005)
14. Mohapatra, S., Dutt, N., Nicolau, A., Venkatasubramanian, N.: Dynamo: A cross-layer framework for end-to-end QoS and energy optimization in mobile handheld devices. IEEE Journal on Selected Areas in Communications 25(4), 722–737 (2007)
15. IFIP 10.4 Working Group on Dependable Computing and Fault Tolerance, `http://www.dependability.org/wg10.4/`
16. Sen, K., Viswanathan, M., Agha, G.: Statistical Model Checking of Black-Box Probabilistic Systems. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 202–215. Springer, Heidelberg (2004)
17. Stehr, M.-O., Kim, M., Talcott, C.: Toward Distributed Declarative Control of Networked Cyber-Physical Systems. In: Yu, Z., Liscano, R., Chen, G., Zhang, D., Zhou, X. (eds.) UIC 2010. LNCS, vol. 6406, pp. 397–413. Springer, Heidelberg (2010)
18. Stehr, M.-O., Talcott, C., Rushby, J., Lincoln, P., Kim, M., Cheung, S., Poggio, A.: Fractionated software for networked cyber-physical systems: Research directions and long-term vision. In: Agha, G., Danvy, O., Meseguer, J. (eds.) Festschrift to the Honor of Carolyn Talcott. LNCS, vol. 7000, pp. 111–144. Springer, Heidelberg (2011)
19. xTune Framework, `http://xtune.ics.uci.edu`
20. Younes, H.: Ymer: A statistical model checker. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 429–433. Springer, Heidelberg (2005), `http://www.tempastic.org/ymer`
21. Yuan, W., Nahrstedt, K.: Energy-efficient soft real-time CPU scheduling for mobile multimedia systems. In: 9th ACM Symposium on Operating Systems Principles (SOSP 2003), pp. 149–163. ACM Press (2003)
22. Yuan, W., Nahrstedt, K., Adve, S.V., Jones, D.L., Kravets, R.H.: Grace-1: Cross-layer adaptation for multimedia quality and battery energy. IEEE Transactions on Mobile Computing 5(7), 799–815 (2006)