ExCAPE Annual Report of Activities

April 2013 to March 2014

1 Overview

ExCAPE proposes a novel approach to software design—*computer augmented program engineering*, in which a programmer and an automated program-synthesis tool collaborate to generate software that meets its specification. A programmer expresses her insights about the design using synthesis artifacts of different kinds such as programs that may contain ambiguities, declarative specifications of high-level requirements, positive and negative examples of desired behaviors, and optimization criteria for selecting among alternative implementations. The synthesis tool composes these different views about the structure and functionality of the system into a unified concrete implementation using a combination of algorithmic techniques such as decision procedures for constraint-satisfaction problems, iterative schemes for abstraction and refinement, and data-driven learning. Our goal is to develop the theory and practice of the proposed approach into a transformative software design paradigm with the promise of a more reliable software at a lower cost. To achieve this goal, our team brings together expertise in theoretical foundations (computer-aided verification, control theory, program analysis), design methodology (human-computer interaction, model-based design, programming environments), and applications (concurrent programming, network protocols, robotics, system architecture).

Progress during the second year of the Expeditions is discussed in this report, which is organized along the research themes of *design methodology* (Section 2), *computational engines* (Section 3), *challenge problems* (Section 4), *tools and infrastructure* (Section 5), and *education and outreach* (Section 6). The list below summarizes and highlights some key advances that would not have been possible without collaboration among the diverse group of PIs.

- **Domain-Specific Synthesis Tools:** ExCAPE researchers are building specialized synthesis tools to facilitate a number of design tasks such as specification of distributed protocols, design of controllers for robots, design of controllers in software-defined networks, and library routines for correct data-structure manipulations. An example of an "end-to-end" tool whose potential has been demonstrated is *CodeHint*. This tool integrates synthesis into a programmer's daily workflow in the commonly used Eclipse development environment. It allows the programmer to use simple assertions as a substitute for concrete code, and dynamically performs a search for correct code based on these assertions. The tool has been evaluated using two user studies that demonstrate significant improvements in programmer productivity.
- **Syntax-Guided Synthesis:** The ExCAPE team observed that the following computational problem is common to a number of emerging synthesis tools: synthesize a (loop-free) program that meets a correctness specification given by a logical constraint as well as a syntactic template. We have formalized this problem as *Syntax-Guided Synthesis*, and standardized the input format. We have built prototype solvers implementing different solution strategies, and collected hundreds of benchmarks. We are organizing a competition of solvers scheduled for Summer 2014. We believe that this effort will both advance the state-of-the-art in computational engines, and facilitate novel applications of program synthesis.
- **Algorithmic Foundations:** ExCAPE researchers have developed an exciting range of novel theoretical tools for formalizing and solving the different versions of the synthesis problem. In particular, we have developed compositional techniques to exploit the design modularity to enhance the scalability of synthesis algorithms. The principle of *counter-example guided inductive synthesis* (CEGIS) has emerged as a unifying methodology that allows the decomposition of the synthesis problem into interacting phases of *learning* candidate programs and *verifying* the correctness of the proposed candidate.
- **Infrastructure for Building Synthesis Tools:** Tools based on constraint solvers (such as verification or synthesis) work best when specialized to a given domain. Domain specific languages (DSLs) are typically embedded, in the form of APIs and interpreters, into modern host languages (for example, JavaScript, Scala or Racket). To facilitate the design of *solver-aided* DSLs for new domains, ExCAPE team has been developing a new host language Rosette. New Solver-Aided DSLs can use Rosette's symbolic virtual machine (SVM) to automate hard programming tasks, including verification, debugging, synthesis, and programming with angelic oracles. Rosette provides a unifying infrastructure for our design methodology, and has hosted several new SDSLs, including imperative SDSLs for data-parallel and spatial programming; a functional SDSL for specifying executable semantics of secure stack machines; and a declarative SDSL for web scraping by example.

- **Integrated System Design for Robotics:** In the robotics domain, ExCAPE team's focus is on system design at different levels. The range of design problems includes how to give commands to a robot using (natural) structured English, how to achieve coordination among a team of robots to achieve a common goal, how to translate high-level specifications to plans, how to map plans to low-level motion primitives, how to ensure robust behavior in presence of uncertainty, and how to bridge the gap between continuous physical world and discrete software commands. An integrated solution to these problems demands collaboration among researchers with different expertise: we have faciliated such a collaboration via multiple collaborative projects, jointly supervised postdoctoral researchers, and working group meetings, resulting in synergistic advances on multiple fronts.
- **Personalized Education:** With the increasing importance of online education, we have realized that technology for formal verification and synthesis can contribute to online learning by analyzing students' solutions to provide meaningful personalized explanation of their mistakes, and also by automatic grading. We have built such tools for representative topics in three undergraduate courses: introduction to programming, theory of computation, and design of cyber-physical systems. The design of these tools not only uses computational engines built for synthesis by ExCAPE researchers, but also integrates HCI research on user interaction and tool evaluation. These tools are already deployed in MOOCs as well as classrooms, and have been used outside ExCAPE institutions. For example, the tool AutomataTutor was used by 300 undergraduate students at Reykjavik University in Iceland in Spring 2014, and received glowing reviews.
- **Cross-disciplinary Community of Synthesis Researchers:** ExCAPE has been instrumental in cultivating a community of researchers spanning subdisciplines such as programming languages, formal methods, computer-aided design, control theory, and robotics, with a common mission of greater automation for system design. A number of our activities are focused on building bridges across communities and outreach. Examples of such activities include research aimed identifying the synergies and differences between reactive synthesis and supervisory control; special sessions on synthesis at conferences on control, robotics, and cyber-physical systems; summer school at UC Berkeley in June 2013 (90 participants); organization of the annual SYNT workshop co-located with the verification conference CAV; organization of SyGuS-Comp (competition of solvers for syntax-guided synthesis); and joint activities with NSF Expeditions projects CMACS and PPM, DARPA/SRC Research center TerraSwarm, and Austrian project RiSE.

2 Design Methodology

The first research theme, *design methodology*, is aimed at understanding how synthesis can be integrated in the software design process so as to decrease the cognitive load on the designers and the programmers. Key contributions over the past year include (1) development of Rosette aimed at simplifying the task of building domain-specific synthesis tools, (2) the Eclipse pug-in CodeHint that integrates synthesis into a stndard program development environment, (3) integration of temporal-logic based contracts in modular design of cyber-physical systems, (4) theoretical foundations of multi-view modeling that allows different stakeholders to use different modeling notations to describe a system, and (5) an extension of OCaml to allow metaprogramming for domain-specific languages.

Infrastructure for Designing a Synthesizer
Bodik (Berkeley)
The goal of ExCAPE is to exploit advances in constraint solving to make programming easier for experts and more accessible to everyone else. The approach is based on two observations. First, much of everyday programming involves the use of domain-specific languages (DSLs) that are embedded, in the form of APIs and interpreters, into modern host languages (for example, JavaScript, Scala or Racket). Second, productivity tools based on constraint solvers (such as verification or synthesis) work best when specialized to a given domain. Rosette is a new kind of host language, designed for easy creation of DSLs that are equipped with solver-based tools. These Solver-Aided DSLs (SDSLs) use Rosette's symbolic virtual machine (SVM) to automate hard programming tasks, including verification, debugging, synthesis, and programming with angelic oracles. The SVM works by compiling SDSL programs to logical constraints understood by SMT solvers, and then translating the solver's output to counterexamples (in the case of verification), or code snippets (in the case of synthesis and debugging). Rosette has hosted several new SDSLs, including imperative SDSLs for data-parallel and spatial programming; a functional SDSL for specifying executable semantics of secure stack machines; and a declarative SDSL for web scraping by example.

CodeHint
PIs: Bodik (Berkeley) and Hartmann (Berkeley)
The overarching goal of ExCAPE is to advance the way programmers write code by leveraging advances in software synthesis. The CodeHint project demonstrates a particular approach for embedding synthesis into a programmer's daily workflow in a standard IDE. The key research question, as it pertains to design methodology, is how to ensure that programmers can easily write specifications of desired code. Formal specifications have shown to be hard and unnatural in part because the programmer may not know the names of relevant classes, which are needed in the specification. In this work, we improve on the expressiveness of specifications of previous synthesizers. Our key idea is making synthesis dynamic. By searching for the correct code at runtime, we enable specifications that are simple assertions over concrete program state, such as "synthesize code that produces an object with a field whose value is a string that contains the string 'hello' ". Notice that neither the type of the object nor the name of its field need to be named. Our implementation, which we call CodeHint, generates and evaluates code at runtime and hence can synthesize real-world Java code that involves I/O, reflection, native calls, and other advanced language features. Our CodeHint design tool implementation was written as a plug-in to the common Eclipse development environment to demonstrate its applicability to today's software engineering workflows. We have evaluated CodeHint in two user studies and have shown that its algorithms are efficient and that it significantly improves programmer productivity [21].

Library-Based Scalable Refinement Checking for Contract-Based Design
PIs: Sangiovanni-Vincentelli (Berkeley), Tripakis (Berkeley)
Synthesis from Linear Temporal Logic is one of the most promising and successful example of correct-by-construction control software synthesis using formal methods. However the utilization of LTL synthesis to solve larger scale problems is still not feasible because of its time and space complexity. Contract-based design provides a theoretical framework that enables the application of platform based design principles, such as composability and reusability, also in the field of LTL synthesis. Sangiovanni-Vincentelli and Tripakis groups, are developing a framework to synthesize control software using contract-based design. A first result [24] has been the development of a technique to verify properties on system designs. Here each design is realized by composition of contracts, chosen from a pre-characterized library (with a particular emphasis on LTL Assume-Guarantee contracts). Thanks to concepts strongly inherent to contract theory, such as composition and refinement, together with the structured nature of the pre-characterized library, it is possible to reduce the problem size by iteratively finding abstractions of the initial design. With this method, it has been possible to observe a performance improvement in terms of execution time up to two orders of magnitude. This solution is being used in a number of case studies, including cyber-physical system design, where a design is defined composing library elements, and motion planning for robotic applications, where the library-oriented approach allows sharing the same high level contracts among different platforms. In this last scenario, the compositional nature of contracts allows dealing with complex tasks, where each contract describes a particular task. Contracts are then implemented by pre-synthesized controllers, loaded by the robot when needed.

Consistency in Multi-View Modeling
PI: Tripakis (Berkeley)
Modeling all aspects of a complex system within a single model is a difficult, if not impossible, task. Multi-view modeling is a methodology where different aspects of the system are captured by different models, or views. A key question then is consistency: if different views of a system have some degree of overlap, how can we guarantee that they are consistent, i.e., that they do not contradict each other? In this work we formulate this and other basic problems in multi-view modeling within an abstract formal framework. We then instantiate this framework in a discrete, finite-state system setting, and study how some key verification and synthesis problems can be solved in that setting. A key synthesis problem is the synthesis of "witness" systems. Consistency of a set of views $V_1, V_2, \ldots, V_k$, is defined as existence of a witness system S, from which each view V_i is derived by an abstraction function a_i, so that $V_i = a_i(S)$. The consistency verification problem is to check, given a set of views, and corresponding abstraction functions, whether the views are consistent. The synthesis problem is to synthesize a witness system. This work paves the way towards a framework where different aspects of a system can be specified separately, by different views or models (e.g., a functional model, a performance model, an energy model) and the system is synthesized automatically from these view-models [34,39].

Fan: Compile-Time Metaprogramming for OCaml
PI: Zdancewic (Penn)
Fan is a programming language extension that supports metaprogramming — i.e. programs that generate other programs. Fan's novel design supports *delimited* domain-specific languages for lexing, dynamic parsing, quasiquotation, and meta-level type-directed programming. As such, it provides language infrastructure for building synthesis systems.

3 Computational Engines

The research theme of *Computational Engines* explores common algorithmic foundations for formalizing and solving the different versions of the synthesis problem. ExCAPE PIs have advanced the algorithmic foundations for synthesis on multiple fronts including heap-manipulating programs, automatic invariant generation, constraint satisfaction, privacy-preserving communication, probabilistic systems, hybrid systems, robustness for dynamical systems, compositional reasoning, and quantitative analysis. Representative efforts are described in greater details below.

Reactive Synthesis vs. Supervisory Control: Bridging the Gap
PIs: Lafortune (Michigan), Tripakis (Berkeley) and Vardi (Rice)
The first part of our activities is a collaborative effort between several ExCAPE PIs to establish a formal connection between synthesis problems that have been considered, largely separately, in the two research communities of control engineering and formal methods. By making this connection mathematically precise, we aim to bridge the gap between two research areas that tackle similar synthesis problems, but from different angles, and by emphasizing different, and often complementary, aspects. Such a formal bridge will be a source of inspiration for new lines of investigation that will leverage the power of the synthesis techniques that have been developed in these two areas. These results, to appear in [18], cover synthesis problems under full observation and partial controllability. Current efforts are aimed at extending the bridge to synthesis problems where with partial observation.

The second part of our activities concerns the development of a new methodology for the synthesis of safe and non-blocking supervisors for partially-observed discrete-event systems. This methodology is based on the construction of a new structure called the "All Inclusive Controller" (AIC), which is a bipartite transition system that embeds all safe and non-blocking supervisors and thus all controllable and observable solutions to the problem. We have developed a methodology that employs the AIC to synthesize supervisors that provably possess a desired maximality property; this was a long-standing open problem in supervisory control. These new results are applicable to the synthesis of supervisors for a wide class of real world cyber-physical systems with partial observations [52].

Synthesis for Hybrid Systems with Maximal Satisfaction Guarantees
PIs: Kavraki (Rice), Kress-Gazit (Cornell) and Vardi (Rice)
We developed a novel theoretical and computational framework for control synthesis for hybrid systems with general nonlinear continuous dynamics from specifications given as a conjunction of a syntactically co-safe and a syntactically safe linear temporal logic (LTL) formulas. If the specification found unsatisfiable by the hybrid system, the framework computes controls that lead to a user-desired partial satisfaction of the liveness segment of the specification without violating the safety requirements. The approach is based on a synergistic combination of planning layers developed in a prior work. Experiments on a hybrid robotic system with nonlinear dynamics demonstrate that the synthesis framework produces satisfactory behaviors when the LTL specification is not fully achievable.

Synthesis for Nondeterministic Hybrid Systems
PIs: Kavraki (Rice) and Vardi (Rice)
We have initiated an effort to examine the problem of synthesis for nondeterministic hybrid systems with complex continuous dynamics and nondeterministic discrete switching between them from a temporal logic specification. Our intention is to employ sampling-based techniques that can deal with nonlinear continuous dynamics and combine it with game-theoretic approaches to handle discrete nondeterminism. The goal of this project is to compute winning strategies for the system to achieve the specification by viewing nondeterminism as an adversary. We will access the framework on a second-order car-like robot in an environment that includes several unknown terrains where moving over each terrain could affect the motion of the robot. In the long run, we are interested in including such systems in synthesis frameworks, a task that is not possible with available machinery [36,28].

Distribution-Aware Sampling Techniques
PIs: Seshia (Berkeley) and Vardi (Rice)
We designed novel algorithms and developed reference implementations for weighted model counting and distribution-aware sampling of satisfying assignments for a SAT formula. The project was motivated by important applications in synthesis tools. In particular, a distribution-aware sampling used in the underlying solvers for sketch-based synthesis provided guarantees on the convergence to an optimal solution. On the other hand, weighted model counting forms the backbone of computational engines for automatic problem generation in context of MOOCs. Prior work in this area scaled only to small problems in practice, or failed to provide strong theoretical guarantees, or employed a computationally-expensive maximum a posteriori probability (MAP) oracle that assumes prior knowledge of a factored representation of the weight distribution. In the first stage of the project, we contributed two novel algorithms, UniWit and ApproxMC, that provided strong theoretical guarantees and scaled to problems involving thousands of variables. Our model counting algorithm, ApproxMC, is the first scalable approximate model counter to the best of our knowledge. We extended the techniques developed in UniWit and ApproxMC and developed novel algorithms, UniGen and WeightMC, that provide stronger theoretical guarantees and also scales to problems involving hundreds of thousands of variables [9].

Synthesis with Identifiers
PIs: Kress-Gazit (Rice) and Seshia (Berkeley)
We consider the synthesis of reactive systems from temporal specifications with identifiers, which can parametrize the input and output. We give a new specification formalism for this problem, prove that the synthesis problem is undecidable, and give a sound algorithm based on computing a pattern-based abstraction of the original synthesis game. Results are demonstrated on the Robot Waiter challenge problem formalized in Year 1 of the ExCAPE project. This is one of the first results that extends the theory of synthesis from temporal logic to a richer class of specifications and to synthesis of infinite-state systems [19].

Synthesis from Component Libraries
PI: Vardi (Rice)
We have started an investigation into the problem of synthesis from component libraries [30]. Our goal is to construct an interface abstraction through which the library components can interact. Our motivation is that by using interfaces, every component can be considered as a black-box, and thus a system can be synthesized when only the specification of every component described by an LTL formula is available to the designer. Moreover, interfaces allow the use of hierarchies of compositions which does not exist in the current state-of-the-art.

Synthesis of Robust Control Software
PI: Tabuada (UCLA)
According to the IEEE standard glossary of software engineering, robustness is the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environment conditions. Our contribution is a synthesis methodology for robust cyber-physical systems (CPS) based on a notion of robustness for CPS termed input-output dynamical stability [41,42,45]. It captures two intuitive aims of a robust design: bounded disturbances have bounded consequences and the effect of sporadic disturbances disappears as time progresses. Our synthesis framework is based on an abstraction and refinement procedure. We first compute a finite-state abstraction for the physical components that can be used to synthesize a reactive controller enforcing robustness with respect to discrete environments. This preliminary design is then refined to the reactive control software interacting with the physical components. The refinement step is based on control theoretic techniques so as to provide robustness with respect to continuous environments. The resulting design is then robust with respect to both discrete and continuous environments. In order to establish the preservation of robustness through the abstraction and refinement process we developed several new notions of simulation and bisimulation. Future work includes an experimental validation of these ideas in the context of the robotic systems challenge problem. We have already made preliminary steps towards this objective by investigating reactive control software synthesis for robotic systems in collaboration with Kress-Gazit's group. These preliminary investigations resulted in a submitted conference paper, see *Synthesis for Robots with Complex Dynamics* on Section 4.2.

Controller Synthesis for Linear Control Systems and Safe Specifications
PI: Tabuada (UCLA)
We developed a novel algorithm to synthesize controllers enforcing linear temporal logic specifications on discrete-time linear control systems [40]. The central step within this approach is the computation of the maximal controlled invariant set contained in a possibly non-convex safe set. Although it is known how to compute approximations of maximal controlled invariant sets, its exact computation remains an open problem. We provide an algorithm which computes a controlled invariant set that is guaranteed to be an under-approximation of the maximal controlled invariant set. Moreover, we guarantee that our approximation is at least as good as any invariant set whose distance to the boundary of the safe set is lower bounded. The proposed algorithm is founded on the notion of sets adapted to the dynamics and binary decision diagrams. Contrary to most controller synthesis schemes enforcing temporal logic specifications for control systems, we do not compute a discrete abstraction of the continuous dynamics. Instead, we abstract only the part of the continuous dynamics that is relevant for the computation of the maximal controlled invariant set. For this reason we call our approach specification guided. Our preliminary implementation handles up to five continuous dimensions and specifications containing up to 160 predicates defined as polytopes in about 30 minutes with less than 1GB memory. No other approach to control software synthesis currently scales to 5 continuous variables.

Synthesizing Invariants for Program Verification
PI: Parthasarathy (UIUC)
The group of PI Parthasarathy has been working for several years on synthesizing invariants for program verification. When building reliable applications, current techniques require the programmer to annotate the program with inductive invariants to aid verification. Providing these annotations is one of the biggest challenges that we face in making such techniques useful for programmers. The project hence focuses on aiding the programmer by synthesizing these annotations automatically. Intuitively, this kind of synthesis is a particular kind of predicate synthesis (as opposed to function synthesis) and is covered by the general SyGuS format proposed in ExCAPE (see *Syntax-Guided Synthesis* on Section 5). The group has been developing predicate synthesis algorithms using learning. Intuitively, the actual constraints are hidden from the learner, and the learner synthesizes predicates based only on concrete examples with their classification as true/false. Machine learning techniques, which are often very scalable, can then be utilized to build classifiers, which then can be interpreted as logical predicates. For invariant synthesis, it turns out that we need to learn from samples where the classification is not known a priori, but where certain pairs of samples are constrained. We have defined a general learning formalism called ICE (learning using examples, counter-examples, and implications) [22]. We are currently generalizing the competitive solvers we have developed (some based on SMT solvers, some based on machine learning classifiers), to work with predicate synthesis problems formulated in SyGuS, in order to build general backend solvers for SyGuS.

Synthesis of Insertion Functions for Enforcement of Opacity Security Properties
PI: Lafortune (Michigan)
We have investigated a new application domain for reactive synthesis techniques, namely, that of enforcement of security properties in networked systems. Specifically, we have considered the notion of opacity, which captures the ability, or lack thereof, of an intruder to infer a "secret" about a given system, based on its knowledge of the system structure and its observation of the system behavior. In cases where the desired opacity property does not hold, we have developed a novel synthesis methodology for the construction of a suitable interface, in the form of an insertion function, that provably enforces opacity by insertions of additional, fictitious, observable events. Our synthesis methodology is based on the construction of a finite structure that embeds all valid insertion functions. Game-theoretic results on weighted graphs can then be leveraged to synthesize optimal insertion functions when costs are assigned to the inserted events. We have shown how this methodology can be used to enforce privacy of the location of a user of location-based services [51].

Synthesizing Provably Correct Data-Structure Manipulations
PI: Parthasarathy (UIUC) and Solar-Lezama (MIT)
The aim of this project is a marriage of two techniques both developed by ExCAPE researchers: the *natural proofs* technique from program verification, and the *sketching* technique from program synthesis. While natural proofs encodes a set of powerful proof tactics into automatically generated ghost code, it allows to use SKETCH to synthesize provably correct code snippets from scratch. We currently focus on data-structure manipulations and specs written in

the Dryad logic. Technically, we are encoding a symbolic interpreter for Dryad into SKETCH, so that SKETCH can synthesize code snippets that meet given Dryad specs. Ongoing case studies include basic list/tree manipulations.

Resilience to Intermittent Assumption Violations in Reactive Synthesis
ExCAPE Researchers: Ehlers (Berkeley/Cornell), Topcu (Penn)
This project considers synthesis of reactive systems that are robust against intermittent violations of their environment assumptions. Such assumptions are needed to allow many systems that work in a larger context to fulfill their tasks. Yet, due to glitches in hardware or exceptional operating conditions, these assumptions do not always hold in the field. Building on *generalized reactivity(1) synthesis*, a synthesis approach that is well-known to be scalable enough for many practical applications, we show how, starting from a specification that is supported by this synthesis approach, we can modify it in order to use a standard generalized reactivity(1) synthesis procedure to find error-resilient systems. As an added benefit, this approach allows exploring the possible trade-offs in error resilience that a system designer has to make, and to give the designer a list of all Pareto-optimal implementations [20].

Regular String Transformations
PI: Alur (Penn)
Theory of regular languages (of strings, infinite strings, and trees) has proved to be an excellent foundation for *qualitative* verification and synthesis of finite-state systems. The goal of this research thread is to develop a similarly robust foundation for *quantitative* analysis and synthesis. We propose a deterministic model for associating costs with strings that is parameterized by operations of interest (such as addition, scaling, and min), a notion of regularity that provides a yardstick to measure expressiveness, and study decision problems and theoretical properties of resulting classes of cost functions. Our definition of regularity relies on the theory of string-to-tree transducers, and allows associating costs with events that are conditional upon regular properties of future events. Our model of cost register automata allows computation of regular functions using multiple "write-only" registers whose values can be combined using the allowed set of operations. We show that classical shortest-path algorithms as well as algorithms designed for computing discounted costs, can be adopted for solving the min-cost problems for the more general classes of functions specified in our model. Cost register automata with min and increment give a deterministic model that is equivalent to weighted automata, an extensively studied nondeterministic model, and this connection results in new insights and new open problems.

In a recent result, we have developed an algebraic and machine-independent characterization of regular string functions analogous to the definition of regular languages by regular expressions. When the set of values is a commutative monoid (for example, numerical costs with addition), we have proved that every regular function can be constructed from constant functions using the combinators of choice, split sum, and iterated sum, that are analogs of union, concatenation, and Kleene-*, respectively, but enforce unique (or unambiguous) parsing. The structure is more fascinating when the set of values is a non-commutative monoid, for instance, the set of output strings with concatenation operation, which is of particular interest for capturing regular string-to-string transformations for document processing. We proved that the following additional combinators suffice for constructing all regular functions: the left-additive versions of split sum and iterated sum, which allow transformations such as string reversal; sum of functions, which allows transformations such as copying of strings; and function composition, or alternatively, a new concept of chained sum, which allows output values from adjacent blocks to mix. The work on regular string transformations, though not immediately applicable to verification and synthesis problems, is of foundational nature with potential for long-term impact, and has led to several publications in theory conferences [6,2,4].

4 Challenge Problems

To demonstrate the potential viability of the ExCAPE approach, and to guide the foundational research along the most promising directions, this theme focuses on representative *challenge problems*. We report on the progress on the five domains, namely, (1) multicore protocols, (2) networked systems, (3) robotic systems, (4) development of applications for mobile platforms, and (5) personalized education. Note that the fifth theme originated in the efforts of ExCAPE PIs on application of constraint solvers and synthesis tools for automated grading and feedback for assignments in courses on introductory programming and theory of computation. A number of interesting research problems arose during these efforts, and we decided to make this domain as a distinct research sub-theme within challenge problems.

4.1 Multicore Protocols

Hardware communication and coordination protocols are the backbone of today's highly integrated Systems-on-Chip (SoC) designs, which are ubiquitous in mobile and embedded computing platforms. Even when employing the state-of-the-art design practices, designers are still called upon to create the entire design, including the most mundane but error-prone low-level aspects of the design, which arguably leads to tedious design and presence of bugs. The goal of this theme is to explore how synthesis can simplify and improve the design process for multicore protocols.

Synthesizing Finite-state Protocols from Scenarios and Requirements
PIs: Alur (Penn), Martin (Penn) and Tripakis (Berkeley)
Scenarios, or Message Sequence Charts, offer an intuitive way of describing the desired behaviors of a distributed protocol. Recently, we have proposed a new way of specifying finite-state protocols using scenarios: we show that it is possible to automatically derive a distributed implementation from a set of scenarios, augmented with a set of safety and liveness requirements, provided the given scenarios adequately "cover" all the states of the desired implementation. We first derive incomplete state machines from the given scenarios, and then synthesis corresponds to completing the transition relation of individual processes so that the global product meets the specified requirements. This completion problem, in general, has the same complexity, PSPACE, as the verification problem, but unlike the verification problem, is NP-complete for a constant number of processes. We have developed two algorithms for solving the completion problem, one based on a heuristic search in the space of possible completions, and one based on a symbolic encoding using OBDDs. We have shown how classical distributed protocols such as the alternating-bit protocol, can be specified in more intuitive ways using this approach.

This work integrates multiple research themes within ExCAPE: the design methodology allows the user to mix different styles of specifications (scenarios, protocol skeletons, and correctness requirements), and is inspired by the work on program sketching; and the computational back-end uses ideas from counter-example guided inductive synthesis. Future work in this project will focus on deriving communicating Extended Finite-State Machines. This will allow the tool to handle more general protocols, and in particular, we plan to focus on industrial-strength protocols for memory consistency and coherence. To solve the computational synthesis problem in this more general setting, we plan to use the emerging infrastructure for the SyGuS (Syntax-Guided Synthesis) problem.

Failure Avoidance in Concurrent Software
PI: Lafortune (Michigan)
We have been working on the efficient synthesis of control logic for failure avoidance in multi-threaded programs, specifically on: (i) a new synthesis technique for deadlock avoidance that employs SAT solvers coupled with control-theoretic methods for systems modeled with Petri nets; and (ii) a new approach for tackling classes of atomicity violations by developing synthesis techniques for Petri nets subject to regular language specifications [8]. The goal in (i) is to achieve greater scalability than prior approaches developed by PI Lafortune and his co-workers in the context of the Gadara project, by exploiting the power of SAT solvers. The goal in (ii) is to synthesize state-partition-based supervisors for enforcing regular language specifications. This will provide the necessary theoretical foundations for tackling atomicity violations in multi-threaded programs by supervisory control techniques.

Chlorophyll
PI: Bodik (Berkeley)
This project rethinks how to construct compilers. Specifically, we ask whether a compiler based on a sequence of synthesis problems can avoid the need for the classical analyses and transformations, which make compilers laborious to develop. In this project, we developed Chlorophyll, a synthesis-aided programming model and compiler for GreenArrays 144, an extremely minimalist low-power spatial architecture that requires partitioning the program into fragments of no more than 256 instructions and 64 words of data. This processor is 100x more energy efficient than its competitors, but it can currently be programmed only by using a low-level stack-based language. Chlorophyll allows programmers to provide human insight by specifying partial partitioning of data and computation. The compiler relies on synthesis, sidestepping the need to develop classical optimizations, which may be challenging given the unusual architecture. To scale synthesis to real problems, we decompose it into smaller synthesis subproblema-partitioning, layout, and code generation. We show that the synthesized programs are no more than 30% slower than highly optimized expert-written programs and faster than programs produced by a heuristic, non-synthesizing version of our compiler [35]. The students on this project won the Qualcomm Innovation Fellowship.

4.2 Robotic Systems

Design of robotic systems, being complete autonomous machines, involves many facets that are absent in the design of other computerized systems. The robotics theme thus present new challenges for automatic synthesis as envisioned by ExCAPE. Motion planning is a prominent example. ExCAPE researchers have developed an open source motion planning library implementing motions solvers that allow the satisfaction of a high level specification while at the same time ensuring a feasible robot motion plan is produced. We have also researched synthesis of more involved aspects of motion planning e.g. considering partially known environments and coordinating multi-robotic motions. Another challenge exhibited under the robotic theme is bridging the gap between continuous physical world and discrete software commands. We report on progress in these and other frontiers belows.

Synthesis with Costs
PIs: Kress-Gazit (Cornell)
We address the problem of synthesizing optimal robot controllers for adversarial environments. Our main observation is that the quality of a path to a goal has two dimensions: (1) the number of phases in which the robot waits for the environment to perform some actions and (2) the cost of the robot's actions to reach the goal. Our synthesis algorithm can take any prioritization over the possible cost combinations into account, and computes the optimal strategy in a symbolic manner, despite the fact that the action costs can be non-integer [26,25].

Synthesis for Robots with Complex Dynamics
PIs: Kress-Gazit (Cornell) and Tabuada (UCLA)
In one approach to synthesizing a reactive control software for robots with complex dynamics, Kress-Gazit's group developed an automatic construction of low-level controllers that ensure the correct continuous execution of a high-level mission plan. Controllers are generated using trajectory-based verification to produce a set of robust reach tubes which strictly guarantee that the required motions achieve the desired task specification. Reach tubes, computed here by solving a series of sum-of-squares optimization problems, are composed in such a way that all trajectories ensure correct high-level behaviors [16,15]. In a second approach to the problem, Kress-Gazit and Tabuada's groups incorporated into the synthesis process an automatically generated abstraction of the robot dynamics. Thus if the controller can be synthesized, the behavior will be correct. They then examined how to provide revisions to the specifications when the dynamics of the robot cause the specification to be unrealizable [14].

Synthesis for Arbitrary Action Durations
PI: Kress-Gazit (Cornell)
We reason about how to synthesize and execute robot controllers for robots with different actions that are not instantaneous, so that the continuous execution is correct with respect to the specification no matter which action completes first. We have shown how this methods generalizes to multi robot tasks [38,48,37].

Robot Plan Synthesis in Partially-Unknown Environments
PIs: Kavraki (Rice), Kress-Gazit (Cornell) and Vardi (Rice)
The collaboration among the PIs resulted in a temporal logic motion planning framework for robotic systems with complex and nonlinear dynamics in partially-unknown environments [31]. The planner employs a multi-layered synergistic framework that can deal with general robot dynamics and combine it with an iterative planning strategy. The work allows dealing with the unknown environmental restrictions only when they are discovered and without the need to repeat the computation that is related to the temporal logic specification. In addition, the method uses a metric for satisfaction of a specification in planning a trajectory that satisfies the specification as closely as possible in cases in which the discovered constraint in the environment renders the specification unsatisfiable. Case studies on a second-order car-like robot in an office environment with unknown obstacles illustrate the efficacy of the framework. In all the case studies, optimal trajectories with respect to the satisfaction measure of the specification were successfully synthesized.

Compositional Multi-Robot Motion Planning
PIs: Pappas (Penn) and Seshia (Berkeley)
We consider the problem of compositional motion planning for multi-robot systems. Given the runtime behavior of a group of robots, specified using a set of safe LTL properties, the goal is to synthesize the robots motion plan. Our method relies on a library of motion primitives that provides a set of controllers to be used to control the behavior of the robots in different configurations. Using the closed loop behavior of the robots under the action of different controllers, we formulate the motion planning problem as a constraint solving problem and use an off-the-shelf satisfiability modulo theories (SMT) solver to solve the constraints and generate trajectories for the individual robot. Our approach can also be extended to synthesize optimal cost trajectories where optimality is defined with respect to the available motion primitives. Experimental results show that this framework has potential to solve complex motion planning problems in the context of multi-robot systems [43].

Collision Avoidance in Vehicular Systems
PI: Lafortune (Michigan)
We are investigating the problem of supervisory control of vehicular networks for collision avoidance at intersections. Our efforts in the past year have been focused on the case of imperfect measurement in the position of the vehicles. Our approach to this problem is to use abstraction techniques for modeling a set of vehicles as a discrete event system, using controllable and uncontrollable events to model control actions, uncontrolled vehicles, and a disturbance. We have shown that this modeling formalism can also be used to incorporate measurement uncertainty by adding another class of observable but uncontrollable events to model measurement. To deal with the fact that the set of measurements are a subset of a continuous rather than a discrete space, we partitioned the set of continuous measurements into a set of equivalence classes (determined with respect to the spatial discretization of the abstraction), and defined an observable but uncontrollable event corresponding to each equivalence class. The same techniques can be extended to other robotic systems, which typically have to deal with the same complicating factors, namely disturbances, uncontrollable environments, and measurement uncertainty [12,13].

4.3 Networked Systems

Traditional internet routing systems involve precise configuration of many low-level parameters on each network devise. Software-defined networks (SDN) is an emerging approach to computer networking which abstracts away these lower level details from the traffic specification, thus allowing network routing reasoning at a higher abstraction level. Over the past year, we have explored synthesis in both traditional internet routing systems (as proposed in the original ExCAPE proposal), as well as in software-defined networks emerging approach.

Synthesis of Data Center Routing Configurations.
PI: Alur (Penn), Loo (Penn)
Mapping virtual networks to physical networks under bandwidth constraints is a key computational problem for the management of data centers. Recently proposed heuristic strategies for this problem work efficiently, but are not guaranteed to always find an allocation even when one exists. Given that the bandwidth allocation problem is NP-complete, and the state-of-the-art SAT solvers have recently been successfully applied to NP-hard problems in planning and formal verification, the goal of this work is to study whether these SAT solvers can be used to solve the bandwidth allocation problem exactly with acceptable overhead. We investigate alternative ways of encoding the allocation problem, and develop techniques for abstraction and refinement of network graphs for scalability. We conducted experimental comparisons of the proposed encodings with the existing heuristics for typical data-center topologies [53].

Synthesis in Software-Defined Networks (SDN)
PI: Loo (Penn)
We recently started exploring the use of synthesis techniques in the context of SDN configuration. With the tremendous growth of the Internet and the emerging software-defined networks, there is an increasing need for rigorous and scalable network management methods and tool support. In [50] we propose a synthesis approach for managing software-defined networks. We formulate the construction of network control logic as a reactive synthesis problem which is solvable with existing synthesis tools. The key idea is to synthesize a strategy that manages control logic in response to network changes while satisfying some network-wide specification. Finally, we investigate network abstractions

for scalability. For large networks, instead of synthesizing control logic directly, we use its abstraction — a smaller network that simulates its behavior for synthesis, and then implement the synthesized control on the original network while preserving the correctness. By using the so-called simulation relations, we also prove the soundness of this abstraction-based synthesis approach.

Synthesis in Traditional Internet Routing Systems.
PI: Loo (Penn)
The performance of networks using the Internet Protocol (IP) stack is sensitive to the precise configuration of many low-level parameters on each network device. These settings govern the action of dynamic routing protocols, which direct the flow of traffic; in order to make sure that the dynamic protocols all converge to produce some 'optimal' flow, each parameter must be set correctly. Multiple conflicting optimization objectives, nondeterminism, and the need to reason about different failure scenarios make the task particularly complicated. Mere simulation is certainly not enough to understand trade-offs and guarantee desired outcomes. We have been adopting a synthesis-based approach, that provides a fast and flexible approach to analysis of this management task. The idea is to combine logical satisfiability criteria with traditional numeric optimization, to reach a desired traffic flow outcome subject to given constraints on the routing process. The method can then be used to probe parameter sensitivity, trade-offs in the selection of optimization goals, resilience to failure, and so forth. The theory is underpinned by a rigorous abstraction of the convergence of distributed asynchronous, message-passing protocols, and is therefore generalizable to other scenarios. Our resulting hybrid engine is faster than either a purely logical or purely numeric alternative, making it potentially feasible for interactive production use [23]. We have also developed a declarative domain specific language (DSL) used for programming secure Internet routing protocols [**?**,10]. Over the next year, we plan to use this DSL as a basis for synthesizing software-defined networking protocols. To handle potential state explosion problem in synthesis, we have developed an automated tool that allows us to reduce a large Internet graph to a smaller one for analysis [49].

Synthesizing Network Updates
PI: Parthasarathy (UIUC)
The group of PI Parthasarathy has made significant progress in synthesizing network updates. Given a current network N and a specification of what the new network should satisfy, the problem is to synthesize the new network N'. Specifications of interest include properties such as (a) all packets in a class P of network packets should get delivered to a different node than where they are going now, (b) all packets in a class P should go to their current destinations, but should go through some firewall, etc. Furthermore, in order for the network N' to be acceptable, we require that N' be obtained from N using a minimal number of changes.

The space of packets is extremely large in any realistic network, and dealing with this state-space, especially in a synthesis setting, is extremely challenging. The group has developed a synthesis approach that uses novel techniques involving abstractions, static analysis, and SMT solvers, to overcome these difficulties, and built a prototype tool that can synthesize network updates. The key idea is to abstract packets into packet classes symbolically, where packet classes capture equivalence classes of packets that take the same path through the current network N. This is followed by a static pruning of the network to obtain the relevant core of the network that will be affected by the update. An abstract model of the network and the packets it processes is then built, where all header information gets hidden, resulting in a significantly simpler model than the original network. Finally, synthesis is solved using standard SMT solvers where constraints are modeled using uninterpreted functions over a theory of sets.

Synthesizing SDN controllers
PI: Parthasarathy (UIUC)
The group of PI Parthasarathy is also investigating the problem of synthesizing controllers for software-defined networks, where the software that instructs the various nodes of what to do when a message arrives, is completely synthesized for a given specification of what the network must do. They are currently building a DSL for such controllers and co-designing the synthesis problem for such controllers.

4.4 Programming for Mobile Apps

Smart-phones and tablets have recently entered our lives and have gained popularity in an unprecedented rapidness. So has the variety and amount of application that run on mobile devices. Programming for mobile devices encounters

some obstacles that are absent in programming for traditional computers. For instance, mobile operating systems such as Andorid and iOS are updated in a frequent manner, requiring the program and its analysis be robust to such changes. Another example has to do with visualization, as applications run both on very small screens and on a variety of different sizes screen, each phenomenon incurring its own set of difficulties. These, and other obstacles unique to mobile apps, offer an opportunity to leverage ExCAPE synthesis solutions to improve productivity of programmers on mobile platforms.

Identifying Bugs in Mobile Applications
PIs: Foster (Maryland) and Solar-Lezama (MIT)
PIs Foster and Solar-Lezama have been working to apply synthesis to the problem of identifying bugs in mobile applications. The challenge in finding bugs in such applications is that symbolic execution, the workhorse of modern bug-finding techniques, has trouble scaling to the complex code that makes up a mobile platform such as Android. The traditional solution to this problem has been to create models of the framework that expose only those aspects of the platform that are necessary to analyze an application. Our goal is to use synthesis to produce such models from traces of the interaction between the application and the platform together with high-level input provided by the user. As a starting point, we have focused on synthesizing models involving the observer pattern, a design pattern pervasive not just in Android, but also in many UI frameworks such as swing. We have developed a prototype tool called *Pasket* (for ***Pa****ttern* ***Sket****cher*) that synthesizes executable models of observers from three artifacts provided by the user: 1. An annotated API of the framework we want to model. The API tells Pasket which classes and methods need to be synthesized; what their types are; and which of them may participate in the observer pattern. 2. Sample applications that exercise the framework in similar ways as the target programs we eventually want to analyze. 3. Auxiliary models that are needed to analyze the tutorial program but that do not involve the observer pattern. Over time, as we are able to synthesize more code we expect the size of this auxiliary information to decrease. As a first step, we decided to try Pasket on Swing, a UI framework that is simpler than Android but shares many of the same design patterns. Using Pasket, we successfully synthesized models of the Swing framework involving JButton and JComboBox. Synthesis took less than two minutes, and the resulting code worked correctly under Java PathFinder (JPF), a Java symbolic executor. We also found that, while JPF includes models of Swing that the JPF developers wrote by hand, they are insufficient to do the same symbolic execution because they neglected models of two methods; this shows some of the benefits of automatic model synthesis. In addition to helping us find bugs in existing Android applications, these models will be crucial in helping us synthesize applications that correctly use the Android platform.

Ringer
PI: Bodik (Berkeley)
This project develops a programming-by-demonstration system for making browsers more useful by avoiding repetitive tasks. Avoiding repetitive tasks, such as scraping data from the web, is especially important on the phone form factor, where screen are smaller and the browsers load pages more slowly than on laptops. Our system, called Ringer, records user's interaction with the browser and uses this demonstration to synthesize a script program that replays this demonstration. The script can also be generalized to repeat the process on related data. For example, the user's demonstration may select one item from the web page (eg, the name of a school) and then enter the school name into another web site that returns the test scores for this school. The script can be generalized under user guidance to repeat the process for all schools on the first web site.

To provide a compelling case study, we have started collaboration with the School of Social Welfare at Berkeley . Our goal is to help social scientists collect data and make inferences form them. From a computer scientist's perspective, the computational challenges faced by social welfare differ from the challenges addressed in big-data research agendas, presenting unique opportunities. First, key sources of data is not readily available in extant databases and must be collected by crawling, for instance, school web sites, parent-teacher meetings, court records, newspaper stories on local DUI incidents and child maltreatment, community photographs, and other sources that together facilitate inference about, say, the state of order or disorder in a neighborhood. In our research we are building tools that social scientists can guide, without any programming, to collect diverse data of interest, on a large scale, and longitudinally. Second, the computer programming needed to join these data sources is more advanced than the typical expertise of a social scientist with R or Python. Rather than require this expertise, we will work on analyzing data by demonstration. Finally, to facilitate model inference, we will allow scientists to express skeletons of models that will be completed with machine learning and software synthesis.

Quicksilver
PI: Bodik (Berkeley)
This project developed a synthesizer of relational queries for data stored in spreadsheets. The specification is a demonstration from the end-user who, by interacting with the synthesizer, gives examples of outputs (both positive and negative). In additional to allowing non-programmers to create complex queries with joins, the synthesizer simplifies work with data on mobile platforms with small displays and tiny keyboards. We have demonstrated the tool on a tablet Android tablet. This work was released as an MS thesis [29] and will continue in the context of developing big-data tools for social scientists.

Parallel Web Browser Technology
PI: Bodik (Berkeley)
This project develops components of fast, parallel web browser. The motivation to accelerate web browsers is to make them fast enough to serve as application platforms on mobile phones; currently, phone apps are developed with iOS or Android, and browsers would make apps portable across phone ecosystems. Recently, in [32], we focused on web document layout, a problem that includes data visualization. We examined how to synthesize a parallel schedule of traversals over document trees. Our document layout programs are declaratively specified as attribute grammars and parallel traversals for GPUs and multicore CPUs are synthesized. To allow the programmer to control the schedule, we defined a declarative language of schedules where programmers may sketch any part of the schedule and the synthesizer will complete the rest. For the same motivation, the synthesizer answers debugging queries about if and how schedules may be completed and enumerates scheduling choices for autotuning .

4.5 Personalized Education

With the increasing importance of online education, we have realized that technology for formal verification and synthesis can contribute to online learning by analyzing students' solutions to provide meaningful, personalized explanation of their mistakes, and also by automatic grading. We have built such tools for representative topics in three undergraduate courses: introduction to programming, theory of computation, and design of cyber-physical systems. The design of these tools not only uses computational engines built for synthesis by ExCAPE researchers, but also integrates HCI research on user interaction and tool evaluation.

AutomataTutor
PIs: Alur (Penn) and Hartmann (Berkeley)
Personalized Education, has emerged as a new challenge area for ExCAPE. Key problems where synthesis can make contributions include automatic grading and provision of automatic feedback. The goal of automatic feedback is to provide a meaningful explanation of students' mistakes. We focus on providing feedback for constructing a deterministic finite automaton (DFA) that accepts strings that match a described pattern [3]. We have developed novel ways of automatically computing alternative hints. Our tool, AutomataTutor, can suggest edits to the student's DFA, highlight phrases from the problem that the student may have misunderstood, or produce a description of the set of all incorrectly handled strings. We also contribute a rigorous evaluation of feedback with 377 students in two university courses. We find that providing either counterexamples or hints is judged as helpful, increases student perseverance, and can improve problem completion time. Second, both strategies have particular strengths and weaknesses, and the preferred choice of feedback depends on the type of mistake and the student. AutomataTutor was used in undergraduate course on Theory of Computation at the following universities: University of Pennsylvania, Fall 2013, 160 students; University of Illinois at Urbana-Champaig, Fall 2013, 180 students; Reykjavik University, Iceland, Spring 2014, 300 students. The students' response to the tool has been enthusiastic.

Auto-Grading Virtual Laboratories in Cyber-Physical Systems
PI: Seshia (Berkeley) and Hartmann (Berkeley)
We have designed an automatic grader for a laboratory in the area of cyber-physical systems [17]. The core technical problem involves parameter synthesis for temporal testers, where the temporal testers are represented in parameterized signal temporal logic. We have demonstrated the effectiveness of our auto-grader on a large data set obtained from an actual on-campus lab course. This work lays a foundation for lab-based online courses in a range of engineering disciplines, including especially Robotics.

Autograder
PI: Solar-Lezama (MIT) and Hartmann (Berkeley)
We are working on a new tool, called *Autograder*, for automatically providing feedback for introductory programming problems [11]. This work comes under the ExCAPE vision on transforming online education. Providing quality feedback to students is already a major problem in classroom courses, but with the advent of MOOCs, this problem has grown exponentially bigger. In order to use Autograder, the instructor only needs to provide a reference implementation of the assignment and an error model consisting of potential corrections to errors that students typically make for the given assignment. Using this information, Autograder derives minimal corrections to student's incorrect solutions, providing them with a measure of exactly how incorrect their solution was, as well as feedback about what they did wrong. We introduce a simple language for describing error models in terms of correction rules, and formally define a rule-directed translation strategy that reduces the problem of finding minimal corrections in an incorrect program to the problem of synthesizing a correct program from a sketch. We have evaluated our system on thousands of real student attempts obtained from the Introduction to Programming course both at MIT (6.00) and at MITx (6.00x). Our results show that relatively simple error models can provide feedback on 64% of all incorrect submissions in our benchmark set.

Teaching Induction with Functional Programming and a Proof Assistant
PI: Zdancewic (Penn) and Hartmann (Berkeley)
This work explores the potential for using functional programming and interactive proof assistants for teaching the concepts of mathematical induction. The resulting proof-of-concept web-based tutorial has students write small functional programs and prove correctness properties using inductive reasoning [33]. In the long run, we anticipate being able to synthesize programming and proving exercises, as well as provide interactive feedback to help students learn the concepts.

4.6 Emerging Directions

Even after adding the application domains of programming for mobile platforms and personalized education, ExCAPE's researchers keep finding new directions in which synthesis technology à la ExCAPE can help. We report on two such emerging directions below. We find the application of formal automatic synthesis for biological models to be of utmost importance as currently, at a broad, biology suffers from lack of rigorous models, mainly since traditional formal reasoning requires strong mathematical background. We feel that ExCAPE-type synthesis tools following approaches such as programming by demonstration can help propagate some more rigorous mathematical reasoning to the area of biology.

Synthesis of Biological Models.
PI: Bodik (Berkeley)
This project uses synthesis to infer executable models of stem cells, using wet-lab experiments as the specification. Fundamentally, this is programming by demonstration, with the experiments playing the role of demonstration and the models being synthesized programs. This work has led to development of new algorithms for synthesis of concurrent programs. The developed algorithms reduce the synthesis problem to three communicating SAT solvers [27,**?**,44].

Synthesizing Language Translators Guided by Algebraic Data Types
PI: Solar-Lezama (MIT)
Our goal is to develop algorithms to learn a language translator through given samples. We represent the input/output sample programs as values of some algebraic data types, and use SKETCH as the synthesis engine. The underlying challenge is how to handle algebraic data types and do pattern matching in SKETCH. Preliminary results show that our algorithms can successfully learn a desugarer for a toy language in a couple of minutes. Ongoing work includes optimizing the algorithms and synthesizing translators for real-world languages.

5 Tools and Infrastructure

The goal of this research theme is to build open-source tools and design environments, and evaluate them for both computational performance and usability. We have reported on many tools in Sections 2, 3 and 4 (Fan, CodeHint,

Pasket, Ringer, Quicksilver, Chlorophyll, AutomataTutor, Autograder and more). The report in this section focuses on tools that provide an infrastructure to a common theme for a divergence of applications and/or divergence of solvers. We have such infrastructure tools both in the design methodology theme as well as the computational engines theme. Together they may be combined to provide both front-end and back-end engines. In addition we have such infrastructure tool in the theme of challenge problems under robotic systems.

Solver-Aided Languages
PI: Bodik (Berkeley)
This research explores how to construct synthesizers rapidly, in a prototyping fashion. In particular, we explore how to avoid building a compiler from programs to logical constraints, which is typically the hardest part of the engineering. The crucial observation is that each domain where synthesizers can be useful constitutes a relatively narrow domain, and the synthesis problems thus can be described with a domain-specific language (DSL), eg communicating automata in cache coherence protocols. We showed that these DSLs can be extended with high-level constructs that make the underlying solver conveniently accessible to the programmer. We showed that these solver-aided DSLs can be built easily, by embedding into a host language, in the same fashion as regular languages are embedded into host languages, such as Scala. The embedding of SDSLs is described in [47]. In [46], we show how to implement the special solver-aided host languages. We believe that these results will make it much easier to construct synthesizers.

Solver-aided domain-specific languages (SDSLs) are emerging as a new class of high productivity programming systems. They ease the construction of programs by using SAT and SMT solvers to automate hard programming tasks, such as verification, debugging, synthesis and simulation. But using solvers requires translating programs to logical constraints — a difficult engineering challenge that has limited the availability of SDSLs. We developed a lightweight approach to implementing a solver-aided host language, which frees SDSL designers from having to compile their languages to constraints. New SDSLs are built by standard shallow (library-based) or deep (interpreter-based) embedding into the host language. The host is equipped with a symbolic virtual machine, which compiles only a small subset of the host's constructs to constraints. The remaining language constructs are stripped away with online partial evaluation. This lightweight compilation is made possible by a novel symbolic execution technique that solves the path explosion problem for a large class of programs. We have used it to implement ROSETTE, a new solver-aided language that is host to several new SDSLs, such as a language for synthesizing web scraping programs.

Syntax-Guided Synthesis
PIs: Alur (Penn), Bodik (Berkeley), Seshia (Berkeley) and Solar-Lezama (MIT)
The classical formulation of the program-synthesis problem is to find a program that meets a correctness specification given as a logical formula. Recent work on program synthesis and program optimization illustrates many potential benefits of allowing the user to supplement the logical specification with a syntactic template that constrains the space of allowed implementation. The motivation is twofold. First, narrowing the space of implementations makes the synthesis problem more tractable. Second, providing a specific syntax can potentially lead to better optimizations. In this project we identified the common core of such problems, and formulated it in a logical framework [1]. The input to the syntax-guided synthesis problem consists of a background theory, a semantic correctness specification for the desired program given by a logical formula, and a syntactic set of candidate implementations given by a grammar. The computational problem then is to find an implementation from the set of candidate expressions so that it satisfies the specification in the given theory. This paves the way for building generic solvers for such problems, instead of devising a dedicated algorithm for each such problem anew. Based on this we initiated a solvers competition — *SyGuS-COMP* (for *Syntax-Guided Synthesis Competition*) that will take place as part of the *FLoC Olympic Games*, an event of *Vienna Summer of Logic 2014*. We have made three initial solvers, that can serve as a reference to candidate solvers, publicly available: enumerative, symbolic and stochastic.

Open Motion Planning Library
PI: Kavraki (Rice)
The library is an open source library that implements sampling-based algorithms and co-safe linear temporal logic specifications. The library allows users to input a robot specification and also a temporal formula in the form of a non-deterministic finite automaton. Several motion planners can be used the developed synergistic framework that allow the satisfaction of the specification while at the same time ensuring a feasible robot motion plan is produced. The library allows for robots with complex dynamics.

6 Education and Outreach

Activities aimed at education, outreach, knowledge transfer, and industry collaborations are discussed in this section.

ExCAPE Summer School
Organizers: Bodik (Berkeley), Lafortune (Michigan) and Zdancewic (Penn)
ExCAPE Summer School was held on Berkeley Campus from June 12 to June 15, 2013. The summer school goal was to expose graduate students and junior researchers to new ideas in program synthesis. The school started with three tutorials with hands on sessions:

- *Reactive Synthesis* by Vardi and Ehlers,
- *Synthesizing Programs with Constraint Solvers* by Bodik and Torlak,
- *Synthesis for Cyber-Phisical Systems* by Tabuada and Rungger.

Supporting lectures were given by ExCAPE PIs Tripakis, Seshia, Lafortune, Solar-Lezama. We had three invited speakers:

- Sumit Gulwani (Microsoft Research): *Synthesis for computer-aided education*,
- Richar Murray (Caltech): *Syntehsis for embedded control software* and
- Alexandre Donze (Berkeley): *Requirement Synthesis for Industrial-Scale Control Systems*.

The school attracted 89 attendees, ranging from undergraduate students to faculty, from over twelve different countries. The feedback we received was overwhelming: 80% of the participants ranked the school has *very good* or *excellent*, and the tutorials were rated as *effective* or *extremely effective* by the majority of participants. We plan to hold similar such summer schools on 2015 and 2017.

ExCAPE Meetings
In the reported period we held three main meetings:

- ExCAPE Annual Meeting, Berkeley, June 10-11, 2013,
- NSF Site Visit, Penn, Aug 20, 2013, and
- ExCAPE Spring Meeting, joint with ARiSE, Berkeley, Mar 10-11, 2013.

All meetings where organized with sessions around ExCAPE's themes. In the annual meeting the challenge problems were organized in parallel sessions, and there was a poster session with 22 posters, and a panel session about *Software Design Methodology Research in the Age of Big Data and Flying Robots* with panelists Patrice Godefroid (Microsoft Research), Aarti Gupta (NEC Labs), Moshe Vardi (Rice University), Mark Wegman (IBM Research), Pieter Mosterman (Mathworks). The Spring meeting was joint with the Austrian Society for Rigorous Systems Engineering ARiSE. It consisted of 50 short talks, including the following invited talks

1. Edwawrd Lee (Berkeley): *The TerraSwarm Research Center*
2. Sumit Gulwani (Microsoft): *Inductive Meta-Synthesizers*
3. Patrice Godefroid (Microsoft): *Dynamic Program Verification + Micro Execution*
4. Jyo Deshmukh (Tema Toyota): *Simulation-Guided Formal Analysis*
5. Susmit Jha (Intel): *Adaptive Computing using Automated Synthesis*

the first of which was designated as keynote speaker. In addition the last part of the meeting was dedicated to discussing possible directions for new collaborations. The third meeting was the largest of all three, attended by more than 70 participants, including all ExCAPE PIs, researchers from ARiSE and industry.

ExCAPE Robotics Workshop
Organizers: Kavraki (Rice) and Vardi (Rice)
ExCAPE Robotics Workshop was held on Rice campus from November 20 to 22, 2013. The workshop was designed for the groups involved with the ExCAPE robotics challenge to get together, give tutorials on their computational tools, and have focused discussions. The purpose of the workshop was threefold: (1) hands-on tutorial on synthesis tools for robotic systems, (2) project updates through presentations, and (3) discussions on the possible future directions

and collaborations. On the first day of the workshop, hands-on tool tutorials were given. The tutorials were on Open Motion Planning Library (OMPL) developed by Kavraki's group, Linear Temporal Logic MissiOn Planning (LTLMoP) developed by Kress-Gazit's group, and Pessoa, a tool for embedded control software synthesis, developed by Tabuada's group. The second day of the workshop was dedicated to project updates, discussions, and comments. On the last day, subgroups were formed to further discuss joint projects and explore possible collaborations based on the themes emerged on the previous day. As a result, new collaborations were initiated on five designated topics, each topic explored by different subgroups of the researchers at Cornell, UCLA, Rice, Michigan and Penn.

ExCAPE Webinar

ExCAPE organizes a monthly seminar series over the web. For each lecture, about 50 participants login from all over the country. The seminar series provides an excellent opportunity for ExCAPE investigators and students to learn about different aspects of synthesis on a regular basis. The talk's slides and video are available on ExCAPE's website. The seminars this period have been:

- *Platform-Based Software Synthesis and Verification Using Contracts*, Alberto Sangiovanni-Vincetelli (Berkeley), May 13 2013.
- *Rational Synthesis*, Dana Fisman (Penn), Jul 18, 2013.
- *Syntax Guided Synthesis*, Rajeev Alur (Penn), Sep 9, 2013.
- *Increasing programmer productivity for building reliable programs using annotations and tactic synthesis*, Madhusudan Parthasarathy (UIUC), Oct 7, 2013.
- *Generealized Synchronization Trees*, Rance Cleaveland (Maryland Univ.), Nov 4, 2013.
- *Programming with Millions of Examples*, Eran Yahav (Technion), Dec 2, 2013.
- *Engineering Domain-specific Languages with Formula 2.0*, Ethan Jackson (Microsoft Research), Jan 6, 2014.
- *Modular Reasoning about Heap Paths via Effectively Propositional Formulas*, Sachar Itzhaky (Tel-Aviv Univ.), Jan 27, 2014.
- *Synthesis of Event Insertion Functions for Enforcement of Opacity Security Properties*, Stepahne Lafortune (Michigan Univ.), Feb 3, 2014.
- *The Satisfiability Revolution and the Rise of SMT*, Clark Barrett (NYU), Mar 3, 2014.

Courses, Workshops, Tutorials and alike

- Syntax-guided synthesis has emerged as a unifying framework for formalizing synthesis problems. PI Alur gave a tutorial to explain this problem at 13th Internal Conference on Formal Methods in Computer-AIded Design (FMCAD 2013) and 11th ACM-IEEE International Conference on Formal Methods and Models for Co-design (MEMOCODE 2013). He is also giving a series of lectures on this topic at Marktoberdorf International Summer School on Dependable Software Systems Engineering in August 2014.
- PI Seshia taught a graduate course at Berkeley *CS 294-98: Formal Methods for Engineering Education* which is exploring the application of many techniques devised in ExCAPE to the challenge problem domain of *Personalized Education*.
- PIs Seshia and Tripakis co-taught Berkeley course *EECS 144/244: Fundamental Algorithms for System Modeling, Analysis, and Optimization* which for the first time in Fall 2013 included lectures on the topics of system and program synthesis, given by co-PI Tripakis. The lectures of this course have been recorded and an on-line version of the course is under development.
- PI Solar-Lezama co-organized PLOOC 2013, PLOOC 2014 — 1st and 2nd Workshops on Programming Languages Technology for Massive Open Online Courses, both co-located with PLDI.
- PI Bodik was the PC Chair of ASPLOS 2013, Houston, TX, March 2013.
- PIs Kavraki and Kress-Gazit organized a workshop on Synthesis for Robotics at the annual Robotics: Science and System (RSS 2013) conference in Germany, June 2013.
- PI Kress-Gazit is organizing a workshop on Formal Synthesis in Robotics at the annual Robotics: Science and System (RSS 2014) conference to be held in Berkeley, CA, July 2014. PI Kavraki is participating in the workshop.
- PI Lafortune organized a special session on Software Synthesis at the 2013 American Control Conference, Washington, DC, June 17-19, 2013. ExCAPE was represented by Lafortune, Pappas, and Seshia. Invited speaker: Richard Murray (Caltech).

Contribution to Research and Teaching Resources

- The ROSETTE solver-aided host language has been released to open source after making it accessible to undergraduates and testing it in classroom on 13 synthesizers. The course materials for this course are publicly available.
- *Autograder* provided to the teaching staff of Introduction to Programming course (6.00) at MIT.
- *Autograder* was presented to the edX faculty teaching introduction to programming. An A/B test study on edX for evaluating its usefulness in this online course is planned.
- AutomataTutor is available for any academic institution (and any private individuals) to use at AutomataTutor.com.

Undergraduates and High-school Outreach

- PI Alur is organizing a session titled *Design of Cyber-Physical Systems: From Pacemakers to Driverless Cars and Beyond* in Penn Engneering's *WICS Highschool Day for Girls*. This event attracts about 100 girls from Philadelphia highschools, and the goal of the event is to inspire them to pursue college education in computer science and engineering. The event is scheduled for April 11, 2014.
- PI Solar-Lezama has engaged two high-school students working on the problems of automatically generating Python programming problems using constraint-based synthesis as part of MIT PRIMES program.
- During Summer 2013, Penn undergraduate student Adam Freilich worked with PI Alur on the topic of regular functions. This work has led to a publication (in submission) [5].

Invited Talks

- *Syntax-Guided Synthesis*; Alur; at 13th International Conference on Formal Methods in Computer-Aided Design (FMCAD), Portland, October 2013.
- *Regular Functions*; Alur; at Horizons in TCS: A celebration of Mihalis Yannakakis's 60th Birthday, Princeton, August 2013; 28th ACM/IEEE Symposium on Logic in Computer Science (LICS), New Orleans, June 2013; Journees d'Informatique Fondamentale de Paris Diderot, Paris, April 2013.
- *Eliminating Concurrency Bugs in Multithreaded Software: A New Approach Based on Discrete-Event Control*; Lafortune; at the 34th International Conference on Application and Theory of Petri Nets and Concurrency, June 2013, Milano, Italy; at INRIA-Rennes, France, May 2013.
- *Control and Diagnosis of Discrete Event Systems: Some Recent Trends*; Lafortune; at the Annual Control Symposium held at GE Global Research, Niskayuna, NY, Sep 2013.
- *Modeling Biology with Solver-Aided Languages*; Bodik; colloquium at University of Wisconsin; keynote at 12th International Conference on Generative Programming: Concepts & Experiences (GPCE'13); New Directions in Software Technology for synthetic biology (NDIST 2013); DARPA Living Foundries Review Meeting (synthetic biology); UCLA; EPFL (Switzerland).
- *Synthesis for parallel mobile web browsers*; Bodik; Mozilla Internet Futures Workshop, November 2013; The 26th International Workshop on Languages and Compilers for Parallel Computing (LCPC), Sep 2013; Internet Futures Symposium, Nokia, May 2013; Samsung; Google.
- *Solver-aided languages and Rosette, the symbolic virtual machine*; Torlak; The Viewpoints Institute; Samsung; IFIP study group; Northeastern University; UCLA; Berkeley ; Purdue.
- *Designing Bespoke Interactive Devices*; Hartmann, Microsoft Social Computing Symposium, Accenture Technology Labs, Jan 2014. Northwestern University Segal Seminar, Dec 2013.
- *Some Wicked Problems of Software Design*; Hartmann, Symposium on Wicked Problems in Socio-Ecological Systems, Oct 2013.
- *Fabbing Sensors and Drawing Gestures: Design Tools for Post-PC User Interfaces*; Hartmann, GE Research, Aug 2013. Qualcomm, Santa Clara, Jun 2013. UIUC, Champagne-Urbana, Mar 2013.
- *On A Tower of Abstraction for Biology*; Vardi, Computational and Intergative Biomedical Research Center, Baylor College of Medicine, May 2013.
- *Assertion-Based Dynamic Verification for SystemC*; Vardi, Intel Research and Development Center, Folsom, CA, May 2013.
- *Branching vs. Linear Time: Semantical Perspective*; Vardi, Distinguished Carl Adam Petri Lecture, 34th International Conference on Application and Theory of Petri Nets and Concurrency, Milan, Italy, June 2013.

- *A Logical Revolution*; Vardi, Computer Science seminar, University of Maryland, Apr 2013; Plenary Talk, 25th Academia Europaea Anniversary Conference, Wroclaw, Poland, Sep 2013; Microsoft Research, Redmond, WA, Aug 2013; Keynote Talk, 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, St. Petersburg, Russia, Aug 2013; Distinguished Lecture, Software Engineering Institute, East China Normal University, Shanghai, China, Oct 2013.
- *The Rise and Fall of Linear Temporal Logic*; Vardi, Horizons in TCS: A Celebration of Mihalis Yannakakis's 60th Birthday. Princeton, Aug 2013; Highlights of Logic, Games, and Automata. Paris, France, Sep 2013; Workshop on Software Correctness and Reliability. ETH, Zurich, Switzerland, Oct 2013.
- *From Aristotle to The iPhone*; Vardi, Conference on Logic Across The University Foundations and Applications. Tsinghua University, Beijing, China, Oct 2013.
- *Robot Motion Planning*, Kavraki, Fourth Annual STEM Research Symposium, co-hosted by the Society of Hispanic Professional Engineers, the National Society of Black Engineers, the Empowering Leadership Alliance of Rice University, and the Richard Tapia Center for Excellence and Equity, Houston, TX, Nov 2013.
- *Computing for the Physical World*, Kavraki, Grace Hopper Distinguished Lecture Series, School of Engineering and Applied Sciences, University of Pennsylvania, Philadelphia, PA, Nov 2013.
- *Motion Planning for Complex Systems*, Kavraki, George A. Bekey Distinguished Lecture, Viterbi School of Engineering, University of Southern California, Lost Angeles, CA, Sep 2013.
- *From Robots to Biomolecules: Computing for the Physical World*, Kavraki, Invited Plenary Talk, International Joint Conference on Neural Networks, Dallas, TX, Aug 2013.
- *Temporal Logic Motion Planning for Robots with Complex Dynamics*, Kavraki, 4th Workshop on Formal Methods for Robotics and Automation, Robotics Science and Systems, Berlin, Germany, June 2013.
- *The Open Motion Planning Library*, Kavraki, Workshop on Motion Planning for Mobile Manipulation: State-of-the-art Methods and Tools, International Conference on Robotics and Automation, Karlsruhe, Germany, May 2013.
- *Physical Computing*, Kavraki Center for Computational and Integrative Biomedical Research (CIBR) Seminar Series, Baylor School of Medicine, Houston, TX, Apr 2013.

Outreach to Industry and Governmental Agencies

ExCAPE PIs are collaborating with industrial researchers, and have started discussions with other government organizations to find new avenues for applying the synthesis technology. Representative efforts are listed below, and depicted in Figure 1.

- PI Lafortune is collaborating with researchers in Facebook to develop synthesis techniques for avoiding concurrency bugs.
- PI Bodik is collaborating with Sumit Gulwani (at MSR) and Susan Stone (School of Social Welfare, Berkeley) on the topic of using Ringer to collect social scientist's big data from the web by the means of programming by demonstration embedded in the web browser.
- PI Bodik is collaborating with Mozilla on their new parallel browser, Servo.
- PI Bodik is collaborating with Leo Meyerovich (Graphistry) on real-time large-scale data visualization using synthesized parallel layout engines.
- PI Solar-Lezama is collaborating with a startup Aspiring Minds, that is building grading tool for C and Java in the spirit of Autograder.
- Zdancewic's group has been interacting with Sumit Gulwani (at MSR) on some recent work about type-directed program synthesis.
- PIs Alur and Solar-Lezama are collaborating with Aaron Stump (University of Iowa) on using the StarExec system developed via NSF grant #1058748 for the SyGuS competition.
- PI Bodik is collaborating with Mozilla, Google, Nokia on synthesis of parallel and incremental web browse layout engines.
- PI Bodik is collaborating with GreenArrays Inc. and Qualcomm on the topic of synthesis-based compilation for low-power architectures.
- PI Seshia is collaborating with Jeff Jensen and Andy Chang (National Instruments) to use synthesis techniques for auto-grading virtual lab assignments in the area of cyber-physical systems.
- Sela Mador-Haim (PhD student at Penn) has joined Coverity, a company that develops tools for software analysis.

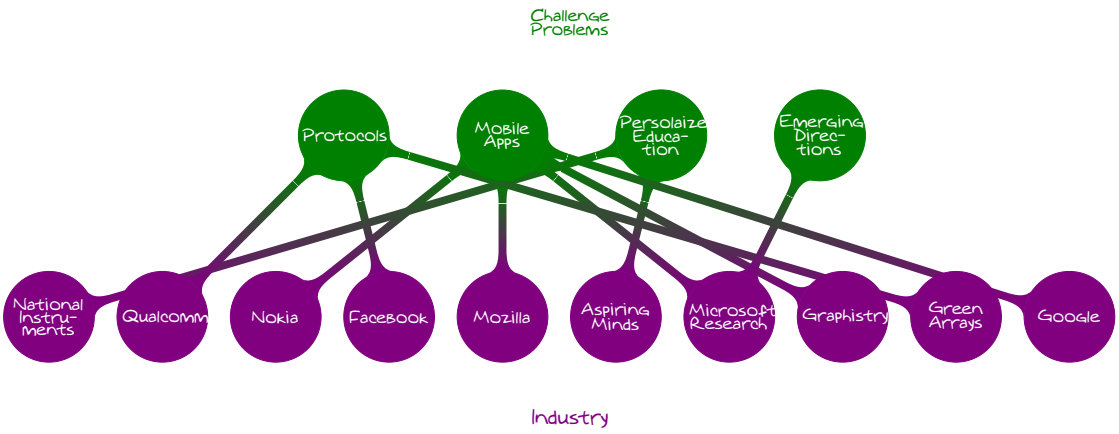

Fig. 1. Outreach to Industry

- A number of Penn students funded by ExCAPE did summer internships: Abhishek Udupa (Intel, Summer 2013); Mukund Raghothaman (Microsoft, Summer 2013); Loris D'Antoni (Microsoft, Summer 2013); Yifei Yuan (NEC, Summer 2013); Salar Moarref (Toyota, Spring 2014); Behnaz Arzani (NEC, Summer 2013).
- PI Alur participated in Mathworks Research Faculty Summit in June 2013, and gave a talk on Software Synthesis.

Summary of the Mentoring Activities Conducted for Postdoc Researchers

- Prof. Ruediger Ehlers, the first postdoctoral researcher of ExCAPE, has obtained a follow-up position of junior research group leader at the University of Bremen in Germany (a position comparable to a non-tenure-track assistant professor position in the US). As co-mentors, PIs Kress-Gazit and Seshia provided him with career counseling and feedback on his job application materials. Prof. Ehlers was closely involved in working with and mentoring graduate students, and gave guest lectures in a graduate course on formal verification and synthesis at Berkeley .
- Dr. Indranil Saha joined ExCAPE in July 2013 and has since been involved in key projects connecting the various institutions involved in the Robotics theme, especially Berkeley and Penn. He has led the development of a novel SMT-based approach to multi-robot motion planning. Moreover, he has also co-led discussions on collaborative projects between PIs Seshia and Tabuada (Berkeley and UCLA). He has been working closely with graduate students at Berkeley and Penn, and will be mentoring course projects relevant to ExCAPE at Berkeley in Spring 2014.
- Dr. Christos Stergiou joined ExCAPE in October 2013 and has since then been leading the research on synthesis of distributed protocols from scenarios and formal specifications, connecting PIs from Penn (Alur, Martin) and Berkeley (Tripakis). This has culminated in a recent submission to CAV 2014. Dr. Stergiou has spent three months in Berkeley and one month in Helsinki, Finland, working closely with PI Tripakis and mentoring graduate students and helping with course preparation beyond research. He is currently at Penn expanding his research agenda to synthesis for cyber-physical systems. He is currently being provided by career counseling by his adviser PIs.
- Dr. Xiaokang Qiu has been working on ExCAPE since August 2013. He has been collaborating with PIs Foster and Solar-Lezama to develop a new methodology to synthesize models of complex platforms like Android, which can be used for both verification and synthesis. Xiaokang has also been starting his own effort within ExCAPE to synthesize provably correct data-structure manipulations in collaboration with PI Madhusidan Parthasarathy. He is also mentoring an undergraduate student Jeevana Priya Inala who has been working with us on the synthesis of language desugaring functions, based on a new approach to encode algebraic datatypes in SKETCH.

7 Management

The ExCAPE Executive Committee consists of Alur (the lead PI), Bodik, Lafortune, Sangiovanni-Vincetelli, and Vardi. This committee has been responsible for ensuring timely progress on research goals. In addition, each ExCAPE theme

has an appointed lead PI (or PIs) guiding research directions on that theme. For the newly added challenge sub-theme on *Personalized Education* we have appointed Hartmann as the lead PI.

In August 2013, we recruited Dana Fisman to be Associate Director of ExCAPE. Dana has a strong background in formal methods: a PhD from Weizmann Institute on temporal logics, and also experience in applications of formal verification in electronic design automation industry. Dana's current research is focused on computational engines for synthesis. She has played a central role in fostering collaborations, organizing ExCAPE webinars and meetings, and putting together reports and presentations. Dana is also a co-organizer of the upcoming competition on Syntax Guided Synthesis.

ExCAPE proposed a novel plan for post-doctoral researchers: each recruit must have at least two mentors from two different institutions, and should foster integrative research activities. Ruediger Ehlers, Xiaokang Qiu, Indranil Saha, and Christos Stergiou have been working on collaborative projects under this scheme (see Summary of mentoring activities). This *rotating postdoc* experiment has worked out very well leading to unexpected and fruitful collaborations. However, we realized that the requirement that the postdoctoral researcher should stay at each host institution for half a year, is onerous and not family-friendly. For the upcoming year, we are going to tweak this arrangement: each researcher must have two mentors from two different institutions, but may choose to stay at one location for the entire duration, and work on a joint project relying on weekly teleconferences and occasional week-long visits.

To ensure that the research directions and the challenge problems lead us towards progress that is meaningful for problems in system design faced by industry, we had set up an industrial advisory board consisting of highly distinguished scientists that represent the range of industries who can potentially benefit from ExCAPE research. The kick-off meeting in June 2012 was attended by John Field (Google), Limor Fix (Intel), Patrice Godefroid (Microsoft), Aarti Gupta (NEC), Himanshu Khurana (Honeywell), Andreas Kuehlmann (Coverity), Pieter Mosterman (Mathworks), Mark Wegman (IBM), and Pamela Zave (AT&T), and we were fortunate to receive valuable feedback from them. Of these, Patrice Godefroid (Microsoft), Aarti Gupta (NEC), Pieter Mosterman (Mathworks), and Mark Wegman (IBM), have continued to stay engaged with the project, and have participated in later meetings, and we are thankful to their feedback (note that lack of participation from other members, while unfortunate, is not unexpected: all these are very busy, with only limited cycles available for academic engagement). As indicated in the previous section, we have also established collaborations with other researchers at Google, NationalInstruments, Qualcomm, Facebook, Microsoft Research, Nokia, AspiringMinds, Graphistry, GreenArrays and Mozilla.

8 Collaborations

Several events have brought all PIs together: meeting and summer school on Berkeley June 2013, NSF site visit at Penn on Aug 2013, spring meeting at Berkeley on March 2014 and the monthly ExCAPE webinars as discussed below. Another big event of excape, was the robotics workshop held at Rice on Nov 2013. Other than that researchers are communicating via regular teleconference meetings and visits to each others labs. Collaborations across themes, disciplines and institutions are depicted in Figures 2,3, and 4.

The following collaborations where made possible due to ExCAPE:

- Ehlers, Lafortune, Tripakis, and Vardi have met several times in person to work on the problem of Bridging the Gap between Reactive Synthesis and Supervisory Control.
- Dallal, Lafortune, and Kress-Gazit started a collaboration to establish a connection between synthesis techniques used by Kress-Gazit and her co-workers and the approach of Dallal and Lafortune for ensuring safety in abstracted models of cyber-physical systems. Case study chosen: collision avoidance. This collaboration started at the robotics workshop Kress-Gazit and Dallal attended and continued with visit of Kress-Gazit at Michigan in January 2014.
- Alur, Martin, Tripakis collaborate on synthesis of protocols from scenarios and specifications with weekly phone calls and in-person meetings.
- Alur, Bodik, Martin, Seshia and Solar-Lezama collaborated on defining a uniform framework for a computational problem common to many recent works that use SMT solvers for synthesizing different programming tasks.
- Alur and Vardi collaborated to put together the report "Theory in Practice for System Design and Verification" (2013) that gives an overview of how theoretical advances in formal verification have influenced industrial practice.
- Seshia and Vardi collaborated on weighted model counting and distribution-aware Sampling.
- Foster and Solar-Lezama have started a collaboration around the project of finding bugs in mobile applications. Solar-Lezama is joining the thesis committee of Jinseong Jeon, who is a student at UMD advised by Jeff Foster. Postdoc Xiaokang Qui is also participating in this effort and is being co-mentored by both PIs.

- Yifei Yuan (advised by Rajeev Alur) completed his WPE-II presentation, and his committee members include Madhu Parthasarathy (UIUC) and Loo Thau Loo (Penn).
- Anduo Wang (advised by Boon Thau Loo) defended her Ph.D. dissertation in Aug 2014. Rajeev Alur (Penn) was on her thesis committee.
- Peter-Michael of Zdancewic's group has worked with Björn Hartmann on developing an evaluation approach for the InductFun tutorial project.
- Pappas, Kumar, and Seshia collaborate on the work on synthesis in the context of multi-robot systems.
- Morteza, postdoc at Kavraki's group visited Tabuada's group at UCLA. They have started a collaboration on sampling-based construction of symbolic models for control systems.
- Ayca Balkan, PhD student at Tabuada's group will visit Moshe's group during the 1st week of March.
- Kavraki, Kress-Gazit and Vardi collaborated on the topic of synthesis for hybrid systems with maximal satisfaction guarantees.
- Kress-Gazit and Paulo's group collaborated on synthesis for robots with complex dynamics.
- Kress-Gazit and Seshia collaborated on the topic of synthesis with identifiers.
- Solar-Lezama collaborated with S. Krishnamurthi et al. (Brown University) on the topic of synthesizing language translators guided by algebraic data types.
- Solar-Lezama collaborated with Parthasarathy on synthesizing provably correct data-structure manipulations.
- Alur and Hartmann collaborated with D. Kini, S. Gulwani and M. Viswanathan on the design, deployment and evaluation of AutomataTutor.
- Hartmann and Bodik collaborated with J. Galenson, P. Reames and K. Sen on the evaluation of CodeHint.
- Hartmann and Solar-Lezama started discussing AutomataTutor evaluation methodology to an autograding system to be deployed in an MIT EdX online course.
- Hartmann and Seshia are developing evaluation approaches for a MOOC that will serve as a testbed for the Personalized Education Challenge Problem.
- Bodik is collaborating with Jasmin Fisher (MSR Cambridge) and Ernest Frankel (MIT) on the topic software synthesis for biology, specifically inference of protein signaling models.
- Bodik is on the PhD committee of Hesam Samimi (UCLA) advised by Jens Palsberg and Todd Millstein, whose thesis was on program synthesis for program repair.
- Bodik visited EPFL to pursue porting Rosette to Scala via LMS; and Sketching combinatorial circuits.

References

1. R. Alur, R. Bodík, G. Juniwal, M. Martin, M. Raghothaman, S. A. Seshia, R. Singh, Armando Solar-Lezama, E. Torlak, and A. Udupa. Syntax-guided synthesis. In *Formal Methods in Computer-Aided Design, FMCAD 2013*, pages 1–17.
2. R. Alur, L. D'Antoni, J. V. Deshmukh, M. Raghothaman, and Y. Yuan. Regular functions and cost register automata. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013*, pages 13–22.
3. R. Alur, L. D'Antoni, S. Gulwani, D. Kini, and M. Viswanathan. Automated grading of DFA constructions. In Francesca Rossi, editor, *IJCAI*. IJCAI/AAAI, 2013.
4. R. Alur, A. Durand-Gasselin, and A. Trivedi. From monadic second-order definable string transformations to transducers. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013*, pages 458–467.
5. R. Alur, A. Freilich, and M. Raghothaman. Regular combinators for string transformations. In *In submission*, 2014.
6. R. Alur and M. Raghothaman. Decision problems for additive regular functions. In F. V. Fomin, R. Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *ICALP (2)*, volume 7966 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 2013.
7. R. Bodík. Modeling biology with solver-aided programming languages. In Jaakko Järvi and Christian Kästner, editors, *GPCE*, pages 1–2. ACM, 2013.
8. A. Butez, Y. Wang, and S. Lafortune. State-partition-based control of discrete event systems for enforcement of regular language specifications. In *IFAC World Congress 2014, to appear*.
9. S. Chakraborty, D. Fremont, K. Meel, S. A. Seshia, and M. Y. Vardi. Distribution-aware sampling and weighted model counting for SAT. In *submitted to AAAI 2014*.
10. C. Chen, L. Jia, H. Xu, C. Luo, W. Zhou, and B. T. Loo. A formal framework for secure routing protocols. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI) (demonstration), 2013*.
11. A. Cheung, A. Solar-Lezama, and S. Madden. Optimizing database-backed applications with query synthesis. In Hans-Juergen Boehm and Cormac Flanagan, editors, *PLDI*, pages 3–14. ACM, 2013.

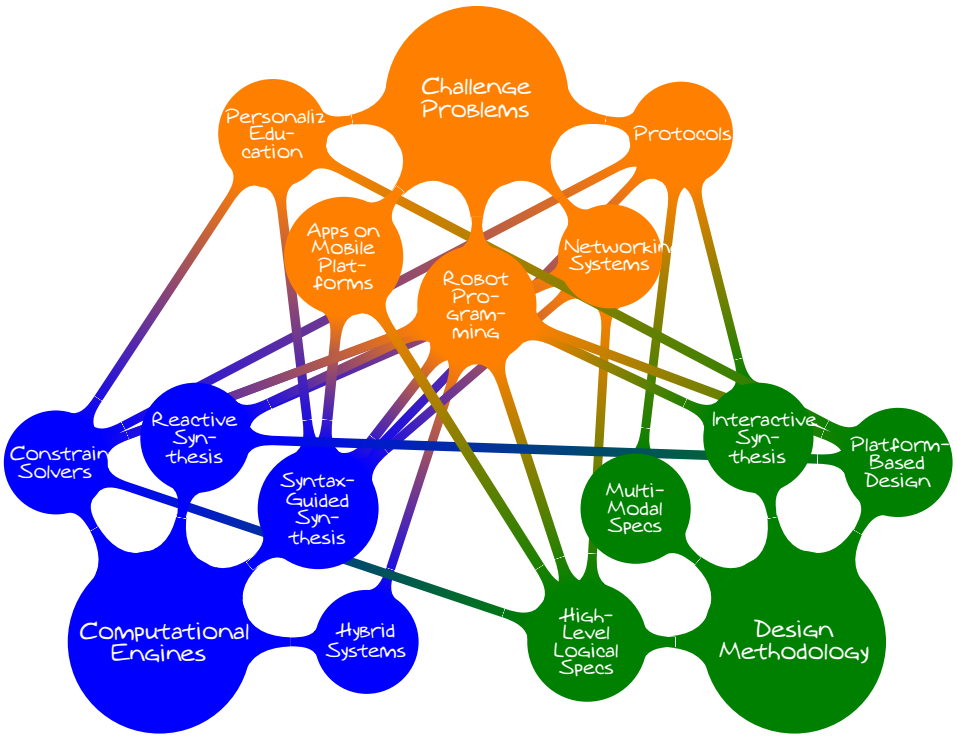

Fig. 2. Collaboration across themes

12. E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune. Supervisory control for collision avoidance in vehicular networks with imperfect measurements. In *Proceedings of the 52nd IEEE Conference on Decision and Control (CDC). December 2013, pp. 6298-6303*.
13. E. Dallal and S. Lafortune. On most permissive observers in dynamic sensor activation problems. In *IEEE Transactions on Automatic Control, To appear.*
14. J. DeCastro, M. Rungger, A. Balkan, R. Ehlers, P. Tabuada, and H. Kress-Gazit. Abstractions and environment restrictions for realizable temporal logic control of complex dynamical systems. In *Proc. of the Robotics: Science and Systems Conference*, 2014.
15. J. A. DeCastro and H. Kress-Gazit. Synthesis of nonlinear continuous controllers for verifiably-correct high-level. *Journal of Reactive Behaviors, under review*.
16. J. A. DeCastro and H. Kress-Gazit. Guaranteeing reactive high-level behaviors for robots with complex dynamics. In *IEEE International Conference on Robotics and Automation, IROS 2013*, pages 749–756.
17. A. Donze, G. Juniwal, J. C. Jensen, and S. A. Seshia. Psgrader: Synthesizing temporal logic testers for auto-grading. In *Submitted to CAV 2014*.
18. R. Ehlers, S. Lafortune, S. Tripakis, and M. Vardi. Bridging the gap between supervisory control and reactive synthesis: Case of full observation and centralized control. In *Proceedings of the 12th International Workshop on Discrete Event Systems WODES 2014, to appear*.
19. R. Ehlers, S. A. Seshia, and H. Kress-Gazit. Synthesis with identifiers. In Kenneth L. McMillan and Xavier Rival, editors, *VMCAI*, volume 8318 of *Lecture Notes in Computer Science*, pages 415–433. Springer, 2014.
20. R. Ehlers, U.Topcu. Resilience to Intermittent Assumption Violations in Reactive Synthesis In *17th International Conference on Hybrid Systems: Computation and Control (HSCC 2014)*.
21. J. Galenson, P. Reames, R. Bodík, B, Hartmann, and K. Sen. Codehint: Dynamic and interactive synthesis of code snippets. In *36th International Conference on Software Engineering, ICSE 2014*.
22. P. Garg, C. Löding, P. Madhusudan, and D. Neider. Learning universally quantified invariants of linear data structures. In Natasha Sharygina and Helmut Veith, editors, *CAV*, volume 8044 of *Lecture Notes in Computer Science*, pages 813–829. Springer, 2013.
23. A. J. T. Gurney, B. Arzani, B. Li, X. Han, R. Guerin, and B.T. Loo. Joint methods for traffic engineering and oscillation repair. In *Submitted to IEEE/ACM Transactions on Networking*.
24. A. Iannopollo, P. Nuzzo, S. Tripakis, and A. Sangiovanni-Vincentelli. Library-based scalable refinement checking for contract-based design. In *Design, Automation and Test in Europe (DATE 2014)*.

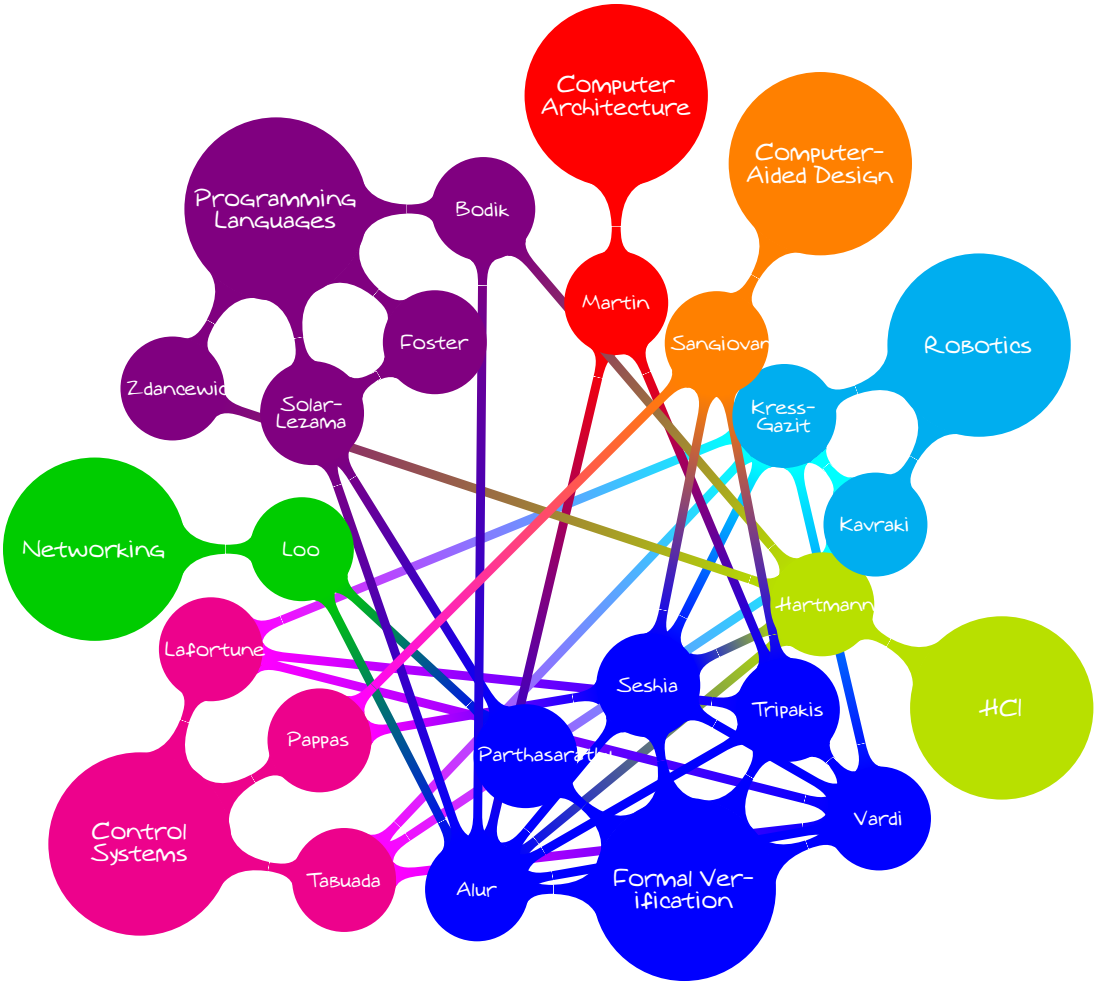

Fig. 3. Collaboration across disciplines

25. G. Jing, R. Ehlers, and H. Kress-Gazit. Shortcut through an evil door: Optimality of correct-by-construction controllers in adversarial environments. In *IEEE International Conference on Robotics and Automation, IROS 2013*, pages 4796–4802.
26. G. Jing and H. Kress-Gazit. Improving the continuous execution of reactive LTL-based controllers. In *ICRA* [**?**], pages 5439–5445.
27. A. Sinan Köksal, Y. Pu, S. Srivastava, R. Bodík, J. Fisher, and N. Piterman. Synthesis of biological models from mutation experiments. In Roberto Giacobazzi and Radhia Cousot, editors, *POPL*, pages 469–482. ACM, 2013.
28. M. Lahijanian, Lydia E. Kavraki, and M. Y. Vardi. A sampling-based strategy planner for nondeterministic hybrid systems. In *IEEE Conference on Robotics and Automation, 2014 (to appear)*.
29. E. Lu and Ras Bodík. Quicksilver: Automatic synthesis of relational queries. Technical report, University of California, Berkeley, Technical Report No. UCB/EECS-2013-68, 2013.
30. Y. Lustig and M. Y. Vardi. Synthesis from component libraries. *STTT*, 15(5-6):603–618, 2013.
31. Matthew R. Maly, M. Lahijanian, Lydia E. Kavraki, H. Kress-Gazit, and M. Y. Vardi. Iterative temporal motion planning for hybrid systems in partially unknown environments. In Calin Belta and Franjo Ivancic, editors, *HSCC*, pages 353–362. ACM, 2013.
32. L. A. Meyerovich, M. E. Torok, E. Atkinson, and R. Bodík. Parallel schedule synthesis for attribute grammars. In Alex Nicolau, Xiaowei Shen, Saman P. Amarasinghe, and Richard Vuduc, editors, *PPOPP*, pages 187–196. ACM, 2013.
33. P-M. Osera and S. Zdancewic. Teaching induction with functional programming and a proof assistant. In *An Education Symposium at SPLASH, SPLASH-E, 2013*.
34. M. Persson, M. Törngren, A. Qamar, J. Westman, M. Biehl, S. Tripakis, H. Vangheluwe, and J. Denil. A Characterization of Integrated Multi-View Modeling for Embedded Systems. In *Proceedings of the 13th ACM & IEEE International Conference on Embedded Software (EMSOFT'13)*, 2013.

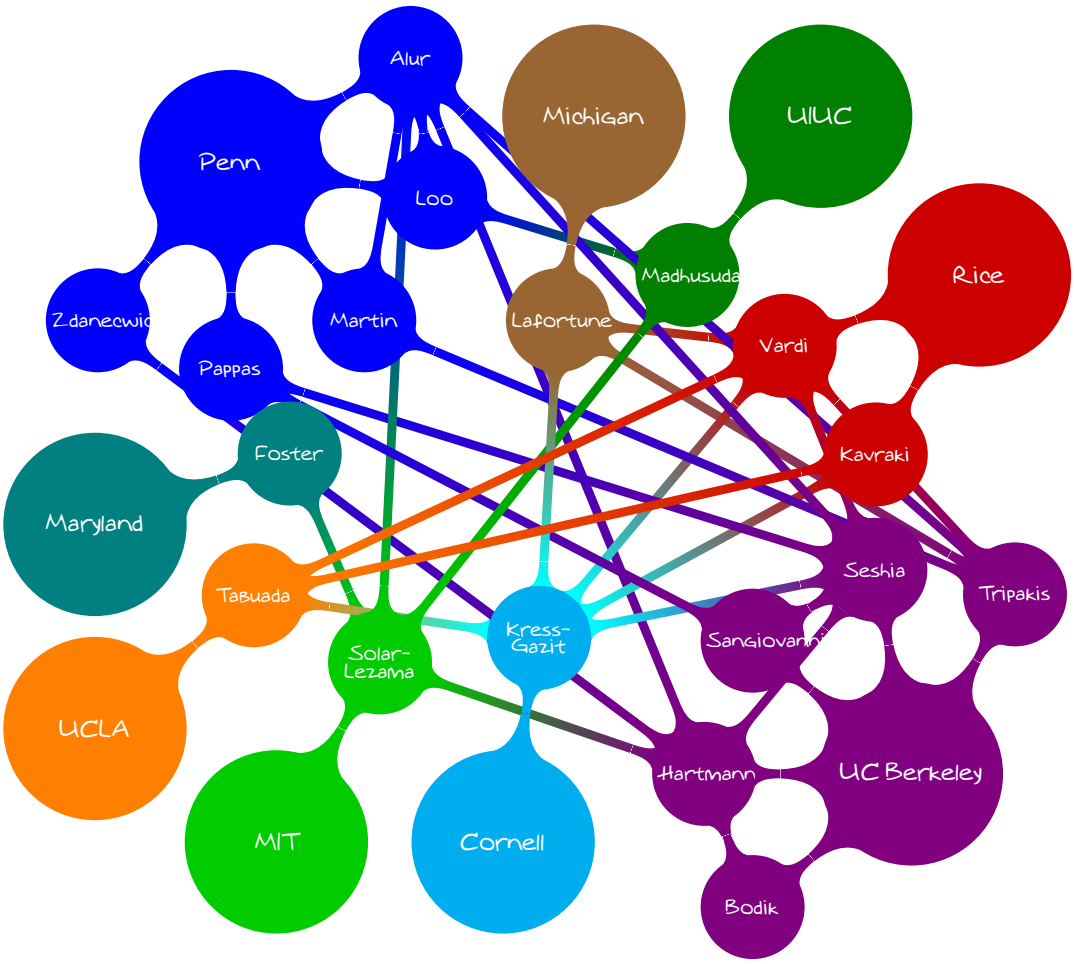

Fig. 4. Collaboration across institutions

35. P M. Phothilimthana, T. Jelves, R. Shah, N. Totla, S. Chasins, and R. Bodík. Chlorophyll: Synthesis-aided compiler for low-power spatial architectures. In *ACM SIGPLAN Conference on Programming Language Design and Implementation PLDI 2014*.
36. E. Plaku, L. E. Kavraki, and M. Y. Vardi. Falsification of LTL safety properties in hybrid systems. *STTT*, 15(4):305–320, 2013.
37. V. Raman and H. Kress-Gazit. Synthesis for multi-robot controllers with interleaved motion. In *IEEE International Conference on Robotics and Automation 2014, to appear*.
38. V. Raman, N. Piterman, and H. Kress-Gazit. Provably correct continuous control for high-level robot behaviors with actions of arbitrary execution durations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, ICRA 2013*, pages 4075–4081.
39. J. Reineke and S. Tripakis. Basic Problems in Multi-View Modeling. In *Tools and Algorithms for the Construction and Analysis of Systems – TACAS 2014*, 2014.
40. M. Rungger, M.J. Mazo, and P. Tabuada. Specification-guided controller synthesis for linear systems and safe linear-time temporal logic. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 333–342. ACM, 2013.
41. M. Rungger and P. Tabuada. A symbolic approach to the design of robust cyber-physical systems. In *Proceedings of the 52nd IEEE Conference on Decision and Control*, 2013.
42. M. Rungger and P. Tabuada. Abstracting and refining robustness for cyber-physical systems. In *Proceedings of the 17th international conference on Hybrid systems: computation and control*. ACM, 2014.
43. I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia. Automated composition of motion primitives for multi-robot systems from safe LTL specifications. In *International Conference on Intelligent Robots and Systems, IROS14*. Submitted.
44. S. Srivastava, T. Hsiau, S. Chasins, J. Kotker, Y-S. Ho, P. Ruan, J. Tsui, S. Hamilton, J. Li, J. C. Anderson, S. A. Seshia, and Rastislav Bodík. Biochemistry as a programming language. In *Proc. Off the Beaten Track (OBT/POPL)*, 2013.

45. P. Tabuada, Sina Y. Caliska, M. Rungger, and Rupak Majumdar. Towards robustness for cyber-physical systems. *IEEE Transactions on Automatic Control*, 2014.
46. E. Torlak and R. Bodík. A lightweight symbolic virtual machine for solver-aided host languages. In *CM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2014*.
47. E. Torlak and R. Bodík. Growing solver-aided languages with rosette. In Antony L. Hosking, Patrick Th. Eugster, and Robert Hirschfeld, editors, *Onward!*, pages 135–152. ACM, 2013.
48. C. Finucane H. Kress-Gazit V. Raman, N. Piterman. Timing semantics for abstraction and execution of synthesized high-level robot control. *IEEE Journal on Transactions on Robotics (IEEE-TRO), in revisions*.
49. A. Wang, A. Gurney, X. Han, J. Cao, B T. Loo, C. Talcott, and A. Scedrov. A reduction-based approach towards scaling up formal analysis of internet configurations. In *33rd Annual IEEE International Conference on Computer Communications (INFOCOM), 2014*.
50. A. Wang, S, Moarref, B T. Loo, U. Topcu, and A. Scedrov. Automated synthesis of reactive controllers for software-defined networks. In *ICNP*, pages 1–6. IEEE, 2013.
51. Y.-C. Wu and S. Lafortune. Synthesis of insertion functions for enforcement of opacity security properties. In *Automatica, To appear*.
52. X. Yin and S. Lafortune. A general approach for synthesis of supervisors for partially-observed discrete-event systems. In *IFAC World Congress 2014, to appear*.
53. Y. Yuan, A. Wang, R. Alur, and B. T. Loo. On the feasibility of automation for bandwidth allocation problems in data centers. In *Formal Methods in Computer-Aided Design, FMCAD 2013*, pages 42–45.