ExCAPE Annual Report of Activities

April 2014 to March 2015

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 (distributed protocols, networked systems, robotics, programming for mobile platforms, and personalized education).

Progress during the third 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. Another example is *NetEgg*: it allows a network operator to specify the desired functionality of a switch using example behaviors. The tool, relying on the computational methods developed by ExCAPE team, automatically generates a SDN implementation of the network policy.
- **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 organized a competition of solvers during FLoC (Federated Logic Conference) in Summer 2014. SyGuSComp is now an annual event, and the second competition will be held in conjunction with CAV in July 2015. 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. 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. New ideas such as *adaptive concretization* and *type-directed synthesis* are enhancing the scalability of CEGIS-based tools. Our work on *theory of regular functions* and *efficient sampling for satisfying assignments* are examples of fundamental advances with broad applications.
- **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. The motion planning software *ComPlan* for distributed mobile robots integrates many of the research ideas with compelling demonstration of its benefits.
- **Computer Aided 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 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 has been already used by over 2000 students from more than ten universities around the world. ExCAPE researchers have also initiated collaboration with experts in education and learning resulting in an upcoming CCC visioning workshop on this topic.
- **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 Berkeley in June 2013 (90 participants) and at MIT in June 2015; 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 project CMACS, 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) continuing development of ROSETTE, a tool for simplifying the task of building domain-specific synthesis tools, which is gaining more and more uses within and beyond ExCAPE, (2) the Eclipse plug-in CodeHint that integrates synthesis into a standard program development environment, (3) Synthesis using adaptive concretization, a new approach to syntax-guided synthesis combining symbolic and explicit search techniques, with the goal to improve the scalability. Additional representative efforts are described in this section.

Infrastructure for Designing a Synthesizer
PI: 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 challenging programming tasks, including verification, debugging, synthesis, and programming with angelic oracles [69]. 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. ROSETTE has been publicly released [9,10] and has been used in the past year in several projects at Berkeley and University of Washington.

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 difficult 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 [43,42]. We presented CodeHint at two workshops in the past year, and started a collaboration with Eddy, a startup building a programmer assistant for Java similar to CodeHint.

Relational Interfaces and Contract Theories
PIs: Sangiovanni-Vincentelli (Berkeley), Tripakis (Berkeley)
The most ambitious goal of the ExCAPE effort is to radically change the way a software system is designed and implemented using synthesis. To guarantee the success of these techniques from small to large scale problems, as can be the design of complex software and cyber-physical systems, defining a compositional synthesis framework is the key factor. In this past year, Sangiovanni-Vincentelli research group kept focusing on the development of a set of Platform-based synthesis and verification tools using libraries of components defined by contracts. To explore possibilities of integrating counterexample-guided synthesis techniques, while maintaining a close cooperation with Tripakis group, a new collaboration with the ExCAPE postdoctoral researcher Christos Stergiou has been started. Together with the ongoing implementation of tools, Sangiovanni-Vincentelli and Tripakis groups contributed to the ExCAPE theoretical foundations [60] investigating the relation between interface theories (specifically, relational interfaces) and assume-guarantee (A/G) contracts. In this work, differences and correspondences between key operators and relations in the two theories (i.e. composition, refinement and conjunction) are analyzed and a natural transformation from relational interfaces to LTL A/G contracts is proposed.

Refinement Calculus for Reactive Systems
PI: Tripakis (Berkeley)
Compositionality is an important theme in system design. In general, compositionality properties allow a designer to decompose a system into a set of components, reason about the components separately, and somehow combine the results to reason about the entire system, formed as the composition of these components. Several compositional frameworks exist for sequential programs and also for reactive systems. The theory of refinement calculus initiated by Dijkstra, Hoare, and others, and developed further by Back, is perhaps the most complete compositional framework to reason about sequential programs. Such a complete framework is missing for the case of reactive systems. In this work, our goal is to develop a refinement calculus for reactive systems. In particular, our aim is to extend existing compositional frameworks for reactive systems to cover: liveness properties (in addition to the standard safety properties which most frameworks are able to handle already); general forms of composition, including feedback. In [63], we proposed such an extension, based on the concept of monotonic *property transformers*. Property transformers are extensions of *predicate transformers* to a reactive system setting. Property transformers are able to express a large class of properties, including liveness properties, expressed for example in a temporal logic such as LTL. Our recent work introduces the property transformer framework, and studies composition and refinement in this framework. The formal framework is implemented in the theorem prover Isabelle, and all our theorems are mechanized in this tool.

Basic Problems in Multi-View Modeling
PI: Tripakis (Berkeley)
In our previous work we formalized and studied basic problems in multi-view modeling: what are views of a system, when is a set of views consistent, can consistency be checked algorithmically, and so on. We provided a generic framework to reason about such problems, and instantiated this framework in the case of discrete systems. The model used to capture discrete systems in our earlier work was a model of *symbolic transition system*, similar to that used in, say, the NuSMV model checker. In recent work, we have been studying extensions of these results to other models of discrete systems, in particular, finite-state automata and languages over finite words such as regular languages, and omega automata and languages over infinite words such as regular omega-languages. This is work in progress, joint with Christos Stergiou (ExCAPE postdoc) and Jan Reineke (Prof. in Saarlandes, Germany).

Type-and-Example-Driven Program Synthesis
PI: Zdancewic (Penn)
We have develped an algorithm for synthesizing recursive functions that process algebraic datatypes. It is founded on proof-theoretic techniques that exploit both type information and input–output examples to prune the search space. The algorithm uses refinement trees, a data structure that succinctly represents constraints on the shape of generated code. We evaluate the algorithm by using a prototype implementation to synthesize more than 40 benchmarks and several non-trivial larger examples. The results demonstrate that the approach meets or outperforms the state-of-the-art for this domain, in terms of synthesis time or attainable size of the generated programs [61].

Synthesizing Language Translators Guided by Algebraic Data Types
PI: Solar-Lezama (MIT)
We developed an algorithm 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. One of the novel aspects of this work is the combination of type inference and counterexample-guided inductive synthesis (CEGIS) in order to support very high-level notations for describing a space of possible implementations. We also consider a new encoding for synthesis problems involving immutable data-structures that significantly improves the scalability. The new approach was evaluated on a set of case studies which most notably include synthesizing desugaring functions for lambda calculus that force the synthesizer to discover Church encodings for pairs and boolean operations, as well as a procedure to generate constraints for type inference. We evaluated the solver behind our system against Z3 and showed that it was faster than Z3 for synthesis problems [45].

Syntheisizing Data-Structure Manipluation Implementations
PI: Solar-Lezama (MIT)
A tool called IMPSKETCH was developed in order derive provably correct implementations of data-structure manipulations involving trees and lists. IMPSKETCH is built on top of sketch and uses the idea of natural proofs of PI Parthasarathy to reduce the problem of synthesizing manipulations that are provably correct for unbounded data-structures into a bounded synthesis problem. With the ImpSketch system, we were able to synthesize provably correct implementations for a number of data-structure manipulations including insertion into binary search trees and sorted lists and doubly linked lists in a matter of minutes.

Synthesis using Adaptive Concretization
PIs: Foster (Maryland) and Solar-Lezama (MIT)
We developed a new approach to syntax-guided synthesis that enhances SKETCH so it can solve the problems generated by PASKET (a tool for synthesizing framework models with design patterns, see Section 4.3). The new approach, termed *adaptive concretization* combines symbolic and explicit search techniques. We observe that in synthesis via symbolic search, the unknowns that parameterize the search space are not all equally important in terms of solving time. Rather, there are certain highly *influential* unknowns that cause synthesis time to increase dramatically. We found that if we use explicit search to concretize influential unknowns with randomly chosen values, and then search symbolically for the remaining unknowns, the synthesis process as a whole can be dramatically faster. Moreover, we can easily parallelize the synthesis procedure by performing different concretizations on different cores. We compared the adaptive containerization to Sketch and the enumerative solver, which won the SyGuS'14 competition, on a small

number of benchmarks from several application domains, and observed that it outperforms both in time to to solve, and the number of solved instances [47].

3 Computational Engines

The research theme of Computational Engines explores common algorithmic foundations for formalizing and solving different versions of the synthesis problem. ExCAPE PIs have advanced the algorithmic foundations for synthesis on multiple fronts including distribution-aware sampling, superotimizing compilers, stochastic systems, privacy-preserving communication, automatic invariant generation, ω-automata learning, and quantitative analysis. Representative efforts are described in greater details below.

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 [29,30]. 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 [28,31,27].

Superoptimizer Construction Framework
PI: Bodik (Berkeley)
Optimal code generation remains an important and challenging problem. Superoptimization is a technique that addresses this problem by searching for an optimal implementation in a target instruction set architecture (ISA) for a given input program. However, building a correct and efficient superoptimizer for a new ISA requires a large amount of effort. We developed GreenThumb, an extensible framework for building superoptimizers. All that is required to extend GreenThumb to a new ISA is to implement an emulator for the new ISA. GreenThumb also provides a new hybrid search technique that combines existing superoptimizer search techniques, which are symbolic search and mutation-based search. Additionally, we designed a new correctness cost function (or fitness function), which is used in mutation-based search, and leads to a more efficient search. To illustrate the flexibility of the framework, we instantiated the GreenThumb framework to two very different ISAs: ARM and GreenArrays. We evaluated the efficiency of the new hybrid search compared to existing approaches on a number of ARM and GreenArrays programs. We found that the hybrid search is the only one that finds an optimal program for all GreenArrays benchmarks. The new cost function increased the number of runs in which the superoptimizer finds an optimal program from 30 to 36 out of 48 runs on ARM benchmarks.

Relentful Strategic Reasoning in Alternating-time Temporal Logic
PI: Vardi (Rice)
Temporal logics are a well-investigated formalism for the specification, verification and synthesis of reactive systems. Within this family, *Alternating-Time Temporal Logic* (ATL) has been introduced as a useful generalization of classical linear and branching-time temporal logics, by allowing temporal operators to be indexed by coalitions of agents. Classically, temporal logics are memoryless: once a path in the computation tree is quantified at a given node, the computation that has led to that node is forgotten. Recently, MCTL has been defined as a memoryful variant of CTL, where path quantification is memoryful. In the context of multi-agent planning, memoryful quantification enables agents to 'relent' and change their goals and strategies depending on the histories of evolutions.

In this project, we introduce *Relentful* ATL(RATL), a kind of temporally memoryful extension of ATL, in which a formula is satisfied at a certain node of a play by taking into account both its future and past [59]. We study the expressive power of RATL, its succinctness, as well as related decision problems. We investigate the relationship between memoryful quantifications and past modalities and prove their equivalence. We also show that both the relentful and the past extensions come without any computational price; indeed, we prove that both the satisfiability and the model-checking problems are 2EXPTIME-complete [59], as for ATL.

The Complexity of Partial-Observation Stochastic Parity Games with Finite-Memory Strategies
PI: Vardi (Rice)
In this project, we study the complexity of two-player partial-observation stochastic games on finite state graphs where player 1 has partial observation and player 2 has perfect observation. We consider the winning conditions that are ω-regular specified as parity objectives. For such games, it was previously shown that the qualitative-analysis problem, which asks whether there is a strategy to ensure that the objective is satisfied with probability 1, is undecidable. Recently, it was shown that the qualitative-analysis problems under finite-memory strategies are decidable in 2EXPTIME. In a more recent work [32], we improve the complexity and show that the qualitative-analysis problems for partial-observation stochastic parity games under finite-memory strategies are EXPTIME-complete. We also establish optimal (exponential) memory bounds for finite-memory strategies required for qualitative analysis.

Reactive Synthesis vs. Supervisory Control: Bridging the Gap
PIs: Lafortune (Michigan), Tripakis (Berkeley) and Vardi (Rice)
This work aims 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 cover synthesis problems under full observation and partial controllability [40]. Current efforts are aimed at extending the bridge to synthesis problems in the context of partial observation.

Another aspect of this effort 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 [78,76,74,75].

Failure Avoidance in Concurrent Software
PI: Lafortune (Michigan)
This effort concerns the efficient synthesis of control logic for failure avoidance in multi-threaded programs. Two directions are pursued: (i) developing a synthesis technique for deadlock avoidance that employs SAT solvers coupled with control-theoretic methods for systems modeled with Petri nets; and (ii) tackling classes of atomicity violations by developing synthesis techniques for Petri nets subject to regular language specifications [26]. 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 [68].

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 [72,73,77].

Alchemist: Synthesis using machine learning
PI: Parthasarathy (UIUC)
This work pursues synthesis of guarded affine functions using an oracle/teacher that answers only equivalence queries. In this setting, in each round, the teacher examines the current hypothesis of the learner and gives a counter-example in terms of an input-output pair where the hypothesis differs from the target function. The learner uses these input-output pairs to refine his hypothesis of the learned guarded affine expression. Our implementation shows that our learner is effective in learning guarded affine expressions, and is more effective than general-purpose synthesis techniques.

Synthesis of State-Observers Resilient to Sensor Attacks
PIs: Sangiovanni-Vincentelli (Berkeley), Seshia (Berkeley) and Tabuada (UCLA)
Motivated by the increasing number of reported attacks on cyber-physical systems (CPS), we are investigating the detection and mitigation of sensor attacks on CPS described by linear dynamics. In particular, our goal is to automatically synthesize algorithms that are able to isolate the sensors that have been compromised by an adversarial attack (detection) while being able to correctly estimate the state of the physical system (mitigation). Our approach combines Satisfiability Modulo Theory (SMT) solvers, convex optimization, and control theory to synthesize state estimators resilient to sensor attacks [66].

Learning Regular Infinitary Languages
ExCAPE Researcher: Fisman (Penn)
A major critique of reactive synthesis, the problem of synthesizing a reactive system from a given temporal logic formula, is that it shifts the problem of implementing a system that adheres to the specification in mind to formulating a temporal logic formula that expresses it. A potential customer of a computerized system may find it hard to specify his requirements by means of a temporal logic formula; instead, he might find it easier to provide good and bad examples of ongoing behaviors (or computations) of the required system, or classify a given computation as good or bad. This is a classical scenario for interactive learning of an unknown language using membership and equivalence queries. The class of languages to be learned consists of languages of infinite words, and is referred to as the class of regular ω-languages. Learning regular ω-languages was a subject of research since the 90s and received several partial answers, until a full solution was obtained in 2008. In this work [25], which is a collaboration with Dana Angluin (Yale), we provide an algorithm for learning any unknown regular ω-language, using a representation which we show may be exponentially smaller than the state-of-the-art.

Synthesis through unification
PI: Alur (Penn)
Given a specification and a set of candidate programs (program space), the program synthesis problem is to find a candidate program that satisfies the specification. We present the synthesis through unification (STUN) approach, which is an extension of the counterexample guided inductive synthesis (CEGIS) approach. In CEGIS, the synthesizer maintains a subset S of inputs and a candidate program P that is correct for S. The synthesizer repeatedly checks if there exists a counterexample input c such that the execution of P is incorrect on c. If so, the synthesizer enlarges S to include c, and picks a program from the program space that is correct for the new set S.

The STUN approach extends CEGIS with the idea that given a program P that is correct for a subset S of inputs, the synthesizer can try to find a program P' that is correct for the rest of the inputs. If P and P' can be unified into a program in the program space, then a solution has been found. We present a generic synthesis procedure based on the STUN approach and specialize it for two different domains (linear arithmetic and bitvectors) by providing the appropriate unification operators. We implemented these specializations in a prototype tool, and we show that the tool often performs significantly better on standard benchmarks than a tool based on a pure CEGIS approach [16].

Symbolic Visibly Pushdown Automata
PI: Alur (Penn)
Nested words model data with both linear and hierarchical structure such as XML documents and program traces. A nested word is a sequence of positions together with a matching relation that connects open tags (calls) with the corresponding close tags (returns). Visibly Pushdown Automata are a restricted class of pushdown automata that process nested words, and have many appealing theoretical properties such as closure under Boolean operations and decidable equivalence. However, like any classical automata models, they are limited to finite alphabets. This limitation is restrictive for practical applications to both XML processing and program trace analysis, where values for individual symbols are usually drawn from an unbounded domain. With this motivation, we introduce Symbolic Visibly Pushdown Automata (SVPA) as an executable model for nested words over infinite alphabets. In this model, transitions are labeled with first-order predicates over the input alphabet, analogous to symbolic automata processing strings over infinite alphabets. A key novelty of SVPAs is the use of binary predicates to model relations between open and close tags in a nested word. We show how SVPAs still enjoy the decidability and closure properties of Visibly Pushdown Automata. We use SVPAs to model XML validation policies and program properties that are not naturally expressible with previous formalisms and provide experimental results on the performance of our implementation [36].

Precise piecewise affine models from input-output data
PI: Alur (Penn)
Formal design and analysis of embedded control software relies on mathematical models of dynamical systems, and such models can be hard to obtain. In this paper, we focus on automatic construction of piecewise affine models from input-output data. Given a set of examples, where each example consists of a d-dimensional real-valued input vector mapped to a real-valued output, we want to compute a set of affine functions that covers all the data points up to a specified degree of accuracy, along with a disjoint partitioning of the space of all inputs defined using a Boolean combination of affine inequalities with one region for each of the learnt functions. While traditional machine learning algorithms such as linear regression can be adapted to learn the set of affine functions, we develop new techniques based on automatic construction of interpolants to derive precise guards defining the desired partitioning corresponding to these functions. We report on a prototype implementation in Matlab, evaluate the performance of our learning algorithm using synthetic data, and also compare it against known techniques using data-sets modeling electronic placement process in pick-and-place machines [24].

DReX: A Declarative Language for Efficiently Evaluating Regular String Transformations
PI: Alur (Penn)
We present DReX, a declarative language that can express all regular string-to string transformations, and can still be efficiently evaluated. The class of regular string transformations has a robust theoretical foundation including multiple characterizations, closure properties, and decidable analysis questions, and admits a number of string operations such as insertion, deletion, substring swap, and reversal. Recent research has led to a characterization of regular string transformations using a primitive set of function combinators analogous to the definition of regular languages using regular expressions. While these combinators form the basis for the language DReX proposed in this paper, our main technical focus is on the complexity of evaluating the output of a DReX program on a given input string. It turns out that the natural evaluation algorithm involves dynamic programming, leading to complexity that is cubic in the length of the input string. Our main contribution is identifying a consistency restriction on the use of combinators in DReX programs, and a single-pass evaluation algorithm for consistent programs with time complexity that is linear in the length of the input string and polynomial in the size of the program. We show that the consistency restriction does not limit the expressiveness, and whether a DReX program is consistent can be checked efficiently. We report on a prototype implementation, and evaluate it using a representative set of text processing tasks [17].

Regular combinators for string transformations
PI: Alur (Penn)
We focus on (partial) functions that map input strings to a monoid such as the set of integers with addition and the set of output strings with concatenation. The notion of regularity for such functions has been defined using two-way finite-state transducers, (one-way) cost register automata, and MSO-definable graph transformations. In this paper, we give an algebraic and machine-independent characterization of this class analogous to the definition of regular languages by regular expressions. When the monoid is commutative, we prove 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. Our main result is for the general case of non-commutative monoids, which is of particular interest for capturing regular string-to-string transformations for document processing. We prove that the following additional combinators suffice for constructing all regular functions: (1) the left-additive versions of split sum and iterated sum, which allow transformations such as string reversal; (2) sum of functions, which allows transformations such as copying of strings; and (3) function composition, or alternatively, a new concept of chained sum, which allows output values from adjacent blocks to mix [18].

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) robotic systems, (2) networked systems, (3) development of applications for mobile platforms, (4) multicore protocols, and (5) personalized education.

4.1 Robotic Systems

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 facilitated such a collaboration via multiple collaborative projects, jointly supervised postdoctoral researchers, and working group meetings, resulting in synergistic advances on multiple fronts.

Plan Synthesis for Robots with Unsatisfiable Specifications
PIs: Kavraki (Rice) and Vardi (Rice)
This study focuses on the problem of robot motion planning from unsatisfiable LTL specifications. A major limiting factor of the use of temporal logics as the specification language in robotics is their Boolean satisfaction condition. To relax this limitation, we introduce a method for quantifying the satisfaction of co-safe LTL specifications, and propose a planner that uses this method to synthesize robot trajectories with the optimal satisfaction value [52]. The method assigns costs to violations of specifications from user-defined proposition costs. These violation costs define a distance to satisfaction and can be computed algorithmically using a weighted automaton. The planner utilizes this automaton and an abstraction of the robotic system to construct a product graph that captures all possible robot trajectories and their distances to satisfaction. Then, a plan with the minimum distance to satisfaction is generated by employing this graph as the high-level planner in a synergistic planning framework previously developed in our group.

Plan Synthesis for Stochastic Systems
PI: Kavraki (Rice)
The problem of planning for robots with action uncertainty from high-level task descriptions is considered in this project. The main objective is to develop a framework that efficiently synthesizes optimal policies for uncertain systems to satisfy a temporal logic specification with maximum probability. To achieve this objective, the evolution of the stochastic system is first abstracted to a discrete, bounded-parameter Markov decision process (BMDP). The procedure is automated by algorithmic discretization of the system's states space and computation of optimal local policies enabling the system to transit between neighboring discrete regions [55,56]. Then, a global control policy over the product of the BMDP abstraction and the DFA representing the specification is generated. Analysis of the framework reveals that as the resolution of the abstraction becomes finer, the global policy converges to optimal [54]. Simulated experiments confirm the theoretical results and illustrate that high-quality policies can be obtained in seconds, orders of magnitude faster than existing methods.

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 [39]. 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 [38].

Synthesis for Arbitrary Action Durations
PI: Kress-Gazit (Cornell)
Most formal approaches for the construction of verifiable high-level robot control use a discrete abstraction of the underlying continuous domain, and make assumptions about the physical execution of actions given a discrete implementation; examples include when actions will complete relative to each other, and possible changes in the robot's environment while it is performing various actions. Relaxing these assumptions gives rise to a number of challenges in the continuous implementation of automatically synthesized hybrid controllers. In these works 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 show that these ideas can be generalized to the multi robot case [65,64].

Synthesis for unknown and dynamically changing Environments
PIs: Kress-Gazit (Cornell), Kavraki (Rice), Vardi (Rice)
Synthesis of correct-by-construction robot controllers from high-level specifications has the advantage of providing guaranteed robot behavior under different environments. Typically, when such controllers are synthesized, assumptions that the user makes about the behavior of the environment, if any, are incorporated into the resulting controller. In practice, however, the environment assumptions may be unknown to the user, thus preventing the application of synthesis. Even if environment assumptions are available, they may not hold during the robot's execution due to modeling errors or unforeseen anomalous operating conditions. In these works we develop algorithms that automatically adjust the robot's behavior, in a provably correct manner, whenever unexpected or new things occur in the environment [53,71,57].

SMT Based Incremental Motion Planning for Multi-Robot Systems
PIs: Pappas (Penn) and Seshia (Berkeley)
We consider the collision-free motion planning problem for a group of robots using a library of motion primitives. One can model the motion planning problem as a boolean combination of linear arithmetic constraints, and solve those constraints using an off-the-shelf SMT solver to generate trajectories for the robots. However, the number of constraints that ensure collision avoidance among the robots grows quadratically with the number of robots, and SMT solvers cannot handle a system with more than a few robots in a reasonable time. To alleviate this scalability problem, we provide an incremental algorithm where we consider the robots one by one for trajectory synthesis, while ensuring that the synthesized trajectory for a robot does not interfere with the trajectories that have already been synthesized. Thus, our incremental motion planning algorithm decomposes a hard SMT solving problem into a number of smaller SMT solving problems. A key problem in our incremental motion planning framework is to create an ordering of the robots so that if the robots have feasible paths to their destinations, the ordering ensures that such trajectories can be synthesized using our incremental motion planning algorithm. We present a priority assignment algorithm to synthesize such an ordering. We apply our method to synthesize trajectories for tens of quadrotors with complex dynamics in a complex environment.

Compositional Multi-Robot Motion Planning from LTL and Timing Specification
PIs: Pappas (Penn) and Seshia (Berkeley)
In this project, we consider the problem of synthesizing trajectories automatically from LTL specification and timing

specification. Our effort in the past year was focused on synthesizing trajectory from specifications written in a subset of LTL, called safe LTL. We wrote a tool named Complan for synthesizing motion plans automatically from safe LTL specification. This year, we have extended our motion planning framework and the tool Complain to deal with any LTL formula. We have demonstrated the capability of our tool on a persistent surveillance scenario. We have also extended our motion planning framework to deal with timing properties. Towards that end, we have introduced bounded time temporal operators on top of LTL. Given a specification in the LTL extended with bounded time temporal operators, our tool Complain is now capable of synthesizing automatically the robot trajectories satisfying the timing constraints. (A video demonstration is available in [3].)

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 [35,34].

Counter-Strategy-Guided Compositional Synthesis for Multi-Agent Systems
PI: Alur (Penn)
We consider the controller synthesis problem for multi-agent systems. We assume that the system model is given as a composition of turn-based game structures where each game structure describes how a controllable agent reacts to its environment. The objective of the system is given as a conjunction of safety properties. The controllers are synthesized compositionally and hierarchically by analyzing the abstractions of the agents' game structures. Abstractions are refined automatically when needed using counter-strategies. We show the potential of our proposed algorithm by applying it to a robot motion planning case study where both abstraction and compositionality help reduce the computational requirements [21].

Pattern-Based Refinement of Interface Specifications in Reactive Synthesis
PI: Alur (Penn)
We consider the problem of synthesizing interface specifications between components in the context of compositional reactive synthesis. Our solution is based on automatic refinement of assumptions and guarantees expressed in linear temporal logic (LTL). We show how behaviors of the environment and the system can be inferred from counter-strategies and strategies, respectively, as formulas in special forms called patterns. Instantiations of patterns are LTL formulas which hold over all runs of such strategies, and are used to refine the specification by adding new input assumptions or output guarantees. We propose three different approaches for compositional refinement of specifications, and demonstrate and compare the methods with examples and a case study [22].

4.2 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.

Scenario-based programming for network policies
PIs: Alur (Penn) Loo (Penn)
This area of work relates to the use of synthesis techniques to address long-standing challenges in implementing network routing protocols correctly. The emergence of programmable interfaces to network controllers offers network operators the flexibility to implement a variety of policies. We propose NetEgg, a programming framework that allows a network operator to specify the desired functionality using example behaviors. Our synthesis algorithm automatically infers the state that needs to be maintained to exhibit the desired behaviors along with the rules for processing network packets and updating the state. We have developed an initial prototype for NetEgg. Our experiments evaluate the proposed framework based on the number of examples needed to specify a variety of policies considered in the literature, the computational requirements of the synthesis algorithm to translate these examples to programs, and the overhead introduced by the generated implementation for processing packets. Our results show that NetEgg can generate implementations that are consistent with the example behaviors, and have performance comparable to equivalent imperative implementations [79,80].

4.3 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.

Ringer: Deterministic Replay of User Actions
PI: Bodik (Berkeley)
One of the crucial building blocks of most end-user programming systems is the ability to deterministically replay a sequence of user actions. In the context of the web, this means being able to replay events such as clicks, menu selections, and form filling actions. Although replaying a program deterministically may sound trivial, knowing when to dispatch events during replay requires overcoming some challenges. For a recorder, the execution is not fully observable. (What event did the user wait for before clicking? In what order were the browser and user events interleaved?) For a replayer, the execution is not fully controllable. (The browser does not provide the ability to reorder events arbitrarily.) And the web page itself may be different each time it is reloaded, whether because of redesigns, obfuscation, or updated data. Deterministic browser replay is impossible in the general case, but we managed to distill from several sources of non-determinism an easy-to-understand framework for which we can (i) describe assumptions under which deterministic replay is guaranteed; and (ii) design successful heuristics for situations that fall outside the replayable boundaries. Our record and replay algorithms are instantiated in a Chrome extension, Ringer, whose source is publicly available [8].

WebCombine: Synthesis of Web-scraping Scripts from Demonstrations
PI: Bodik (Berkeley)
As more and more data appears on the web, programmers and non-programmers alike are increasingly interested in web harvesting. Although many scraping languages have been developed to help programmers with these tasks, most end-user scraping tools have restricted users to scraping datasets in which all cells of a given row appear on a single list page, with no user interaction required. These techniques are very robust and effective, but unfortunately this model cannot accommodate even a task like scraping friends' cell numbers from Facebook, since getting from the list of friends to a given friend's cell number means clicking on a profile page, then an "About" link, and only then scraping. Building on top of our record and replay system, Ringer, we have produced WebCombine, a PBD web scraping system that allows users to scrape data hidden behind complicated user interactions. Users demonstrate how to collect the first row of the dataset, and the tool collects the rest. By demonstrating lists (introducing for loops) and recording interactions (adding to loop bodies), WebCombine users can build and run quite complicated scraping programs, without writing a single line of code. (A video demonstration is available in [13]. The source code publicly available [12].)

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 [58,44], 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. During the last year, we have been developing algorithms for synthesis of incremental layout engine algorithms.

Synthesizing Framework Models with Design Patterns
PIs: Foster (Maryland) and Solar-Lezama (MIT)
This project is concerned with the problem of automatically synthesizing models of frameworks that use design patterns. A tool named PASKET, was developed for this task [46]. PASKET is an example of *multi-modal synthesis* — it consumes several sources of input to perform its work. The first input to PASKET is the framework API, in terms of classes, methods, and types. The second is *logs* gathered by running a *tutorial program* against the actual framework, and recording the call–return sequence between the program and the framework. Third, PASKET also includes internal knowledge of three design patterns: Observer, Accessor, and Singleton.

PASKET's synthesis process then works via what we call *design pattern matching*: instantiating the design patterns to the framework API, such that the resulting model, when run against the same apps, produces the same logs. It relies on several improvements to SKETCH, to use *adaptive concretization* as elaborated on in Section 2.

4.4 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.

Chlorophyll: a synthesis-aided compiler for spatial architectures
PI: Bodik (Berkeley)
Chlorophyll is a synthesis-aided programming model and compiler for the GreenArrays GA144, an extremely minimalist low-power spatial architecture that requires partitioning the program into fragments. The Chlorophyll programming model allows programmers to provide human insight by specifying partial partitioning of data and computation. The Chlorophyll 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 the compilation into smaller synthesis subproblems: partitioning, layout, and code generation [62]. During the last year, Chlorophyll have been extended to generates both the GA144 assembler code and ARM assembler code. In addition, the superoptimizer parts of the Chlorophyll system have been ported to a compiler developed in Qualcomm Research, during an internship by Phitchaya Mangpo Phothilimthana in Summer 2014.

Automatic Completion of Distributed Protocols with Symmetry
PI: Alur (Penn) and Tripakis (Berkeley)
A distributed protocol is typically modeled as a set of communicating processes, where each process is described as an extended state machine along with fairness assumptions, and its correctness is specified using safety and liveness requirements. Designing correct distributed protocols is a challenging task. Aimed at simplifying this task, we allow the designer to leave some of the guards and updates to state variables in the description of extended state machines as unknown functions. The protocol completion problem then is to find interpretations for these unknown functions while

guaranteeing correctness. In many distributed protocols, process behaviors are naturally symmetric, and thus, synthesized expressions are further required to obey symmetry constraints. Our counterexample-guided synthesis algorithm consists of repeatedly invoking two phases. In the first phase, candidates for unknown expressions are generated using the SMT solver Z3. This phase requires carefully orchestrating constraints to enforce the desired symmetry in read/write accesses. In the second phase, the resulting completed protocol is checked for correctness using a custom-built model checker that handles fairness assumptions, safety and liveness requirements, and exploits symmetry. When model checking fails, our tool examines a set of counterexamples to safety/liveness violations to generate constraints on unknown functions that must be satisfied by subsequent completions. For evaluation, we show that our prototype is able to automatically discover some interesting missing details in distributed protocols for mutual exclusion, self stabilization, and cache coherence [23].

Synthesizing Finite-state Protocols from Scenarios and Requirements
PI: Alur (Penn), Martin (Penn), and Tripakis (UC Berkeley)
Scenarios, or Message Sequence Charts, offer an intuitive way of describing the desired behaviors of a distributed protocol. In this paper we propose 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 present an algorithm for solving the completion problem, based on counterexample-guided inductive synthesis. We evaluate the proposed methodology for protocol specification and the effectiveness of the synthesis algorithm using the classical alternating-bit protocol, the VI cache-coherence protocol, and a consensus protocol [20].

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. We have developed novel ways of automatically computing alternative hints. Our tool, AutomataTutor [1], 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 [37]. 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 is now used at a growing number of universities (UIUC, UPenn, UC San Diego, Cambridge, Reykjavik University, University of Liverpool, EPFL, Mahidol University International College, University of Louisiana at Monroe, IIIT Delhi, John Hopkins) and is used by around 2000 students.

Autograding and problem synthesis for hybrid automata construction - CybOrg
PIs: Sangiovanni-Vincentelli (Berkeley) and Seshia (Berkeley)
Providing the ability to generate personalized exercises and feedback to students, synthesis is the critical element for

the development of tools for Massive Open Online Courses (MOOCs). In this regard Sangiovanni-Vincentelli and Seshia groups developed CybOrg, a feedback and problem synthesis tool for cyber-physical system design MOOCs. CybOrg, which interfaces with Ptolemy (http://ptolemy.eecs.berkeley.edu/) and SpaceEx (http://spaceex.imag.fr/), is able to generate a set of personalized problems starting from an instructor's template. For a given problem, a student's solution is a Ptolemy model of a hybrid automaton. Once submitted the solution, the student receives a feedback indicating whether her solution is correct or not. If not, an explanation of the issues is provided. Moreover, the instructor is able to include a set of hints to be provided to students for the most common errors.

Auto-Grading Virtual Laboratories in Cyber-Physical Systems
PI: Seshia (Berkeley)
CPSGrader is an automatic grading and feedback generation tool for laboratory courses developed by PI Seshia's group at the University of California, Berkeley. The core technology in CPSGrader involves the synthesis of temporal logic testers formalized in Signal Temporal Logic (STL) to monitor traces of student solutions for faults [48,49,50]. It is designed to work with any black-box virtual lab simulator or even a real sensor stream. CPSGrader was successfully deployed in a massive open online course (MOOC) on Cyber-Physical Systems taught by PI Seshia on edX in May-June 2014. It was also deployed on campus in the undergraduate Embedded Systems course at UC Berkeley during Fall 2014, and helped instructors deal with a surge in the enrollment for the course. In both deployments, the student feedback was overwhelmingly positive. The software has been publicly released and is currently being deployed to other courses as well.

AutoProf
PI: Solar-Lezama (MIT) and Hartmann (Berkeley)
We have continued our work on *AutoProf*, a tool for automatically providing feedback for introductory programming problems [33]. 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 AutoProf, 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, AutoProf 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. The AutoProf system was integrated into the EdX platform and is being used in a pilot test involving students taking the classroom version of the course.

4.6 Emerging Directions

ExCAPE's researchers keep finding new directions in which synthesis technology à la ExCAPE can help. We report such directions that do not fall in one of our challenge problems here. 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. 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 [41,51].

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 (CodeHint, Pasket, Ringer, WebCombine, Chlorophyll, AutomataTutor, AutoProf 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)
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 continued working on rosette [70,69], 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 written a tutorial and user guide for ROSETTE [10], and publicly released its source code [9]. ROSETTE is now being used in the tools *Chlorophyll*, a synthesis-aided language and compiler for ultra low-power spatial architectures; *SynthCL*, an imperative SDSL based on OpenCL for verification and synthesis of data-parallel programs; *IFC*, a functional SDSL for specifying and verifying executable semantics of secure stack machines, *WebSynth*, a declarative SDSL for web scraping by example; *BagPipe*, a solver-aided implementation of the Border Gateway Protocol for specifying router configurations; and *Neutrons* a solver-aided implementation of the EPICS dataflow language for distributed soft real-time control systems for scientific instruments, such as telescopes and particle accelerators.

Syntax-Guided Synthesis
PIs: Alur (Penn), Bodik (Berkeley), Parthasarathy (UIUC), 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 [15]. 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*) which was held as part of the *FLoC Olympic Games*, an event of *Vienna Summer of Logic, July 2014*. Five solvers competed in the competition, on more than 250 benchmarks we collected from 10 domain categories. Results on the relative performance of these five solvers and the competition results will appear in [14]. The second *SyGuS-COMP* competition is planned to take place on July 2015, as a satellite event of *CAV* and *SYNT*.

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 previously required the input of a temporal formula in the form of a non-deterministic finite automaton. We have enhanced it to receive an input by means of a formula in co-safe LTL, that will be converted to an automaton transparently to the user, and plan to release it in the next few months. We believe that this will make our work much more accessible and attractive to the general user. We have also continued our efforts to continue to support the integration of the library with the Robot Operating System (ROS), and the widely used visualization package OpenRAVE, so that it can be used on a variety of robotic platforms.

6 Education and Outreach

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

ExCAPE Annual Meeting This year's annual meeting was held at Penn, on October 26-28. This was also NSF's site review meeting. The review team members where Ewen Denney (NASA), Bill Griswold (UCSD), John Harrison (Intel), Viktor Kuncak (EPFL), Frank Pfenning, chair (CMU), and John Reppy (U. Chicago). The meeting was organized around ExCAPE themes; each challenge theme was described by one of the PIs, as follows:

1. *Robotic Systems* by Hadas Kress-Gazit (Cornell)
2. *Personalized Education* by Bjoern Hartmann (berkeley)
3. *Multicore & Distributed Protocols* by Stavros Tripakis (Berkeley)
4. *Programming for Mobile Applications* by Jeff Foster (Maryland)
5. *Networked Systems* by Boon Thau Loo (Penn)

Four representative talks were given on Computational Engines and Design Methodology, as follows:

1. *Syntax Guided-Synthesis* by Armando Solar-Lezama (MIT)
2. *SAT sampling* by Moshe Vardi (Rice)
3. *Solver Aided Languages* by Ras Bodik (Berkeley)
4. *Robust Synthesis* by Paulo Tabuada (UCLA)

Overview and wrap up talks where given by the lead PI, Rajeev Alur (Penn). Talks on Collaboration and management, education and outreach and knowledge transfer where given by Dana Fisman (Penn), Steve Zdancewic (Penn), and Stephane Lafortune (Michigan), resp. During Lunch we held a poster session, in which 19 of ExCAPE PhD students and postdocs presented their work.

ExCAPE Summer School The second ExCAPE summer school will take place in June 2015, at MIT campus. The summer school goal is to expose graduate students and junior researchers to new ideas in program synthesis. The school will have the following hands-on tutorials:

– *Reactive Synthesis* by Roderick Bloem (Graz)
– *Inductive learning and constraint solving* by Sanjit Seshia (UC Berkeley)
– *Syntax-Guided Synthesis* by Armando Solar-Lezama (MIT)
– *Probabilistic programming* by Vikash K. Mansinghka (MIT)
– *Synthesis for robotics* by Hadas-Kress Gazit (Cornell) or alternate.

Supporting lectures will be given by Viktor Kuncak (EPFL), Ashish Tiwari (SRI), Martin Vechev (ETH), David Walker (Princeton), and Keith Winstein (Stanford).

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:

– *Synthesis of Network Updates* by Pavol Cerny (Univ. of Colorado Boulder)
– *Automated Program Synthesis at Kestrel Institute* by Cordell Green and Doug Smith (Kestrel Inst.)
– *On-Demand Printable Robots* by Ankur Mehta (MIT)
– *Software Synthesis using Automated Reasoning* by Ruzica Piskac (Yale Univ.)
– *Solving Horn Constraints for Program Verification and Synthesis* by Andrey Rybalchenko (Microsoft Research)
– *Standing on the Shoulders of a Giant: One Person's Experience of Turing's Impact* by David Harel (Weizmann)
– *Parameterized Synthesis* by Roderick Bloem (TU Graz)
– *From Reachability to Temporal Specifications in Game Theory* by Orna Kupferman (Hebrew Univ.)

Contribution to Research and Teaching Resources

- AutomataTutor [1] is now used at a growing number of universities (UIUC, UPenn, UC San Diego, Cambridge, Reykjavik University, University of Liverpool, EPFL, Mahidol University International College, University of Louisiana at Monroe, IIIT Delhi, John Hopkins) and is used by around 2000 students.
- The AutoProf system for automated tutoring was integrated into the EdX platform and is being used in a pilot test involving students taking the classroom version of the course.
- PI Parthasarathy helped build automated problem generation for exams on design and analysis of algorithms [7] in the MOOC platform MEC (Massively Empowered Classrooms), to study and prevent plagiarism.
- PI Alur's textbook *Principles of Cyber-Physical Systems* will be published by MIT Press in April 2015. The textbook is already being used this Spring semester at Penn and at Chalmers University of Technology, Sweden.
- ROSETTE's webpage has launched, featuring a tutorial and user guide [10] (ROSETTE's source code was released last year [9]).
- Chlorophyll source code has been made publicly available [2].
- A new version of the tools DESUMA and UMDES for supervisory control and diagnostics of discrete event systems were released [5,11].
- CPSGrader was publicly released [4] and deployed on campus at Berkeley.

Courses

- PI Solar-Lezama and ExCAPE postdoc Xiaokang Qiu have been developing a new synthesis course involving a lot of ExCAPE research. Overall, the course covers classical work on automata-based synthesis of reactive systems, as well as recent advances in the use of SAT/SMT solvers to synthesize programs. It also covers the use of heuristic search techniques to explore large spaces of candidate programs. The course also discusses different approaches to address the specification problem, ranging from logic-based specification mechanisms to programming by demonstration. The course is graded on the basis of three homework assignments and an open ended final project. The course webpage includes lecture slides and assignments [67].
- PI Kavraki taught a course at Rice University: *Algorithmic Robotics*, which explores algorithmic techniques for motion planning including the ones devised in ExCAPE to the challenge problem domain of robotic systems.
- PI Vardi taught a summer course on *The Automata-Theoretic Approach to Design and Verification*, 12th LASER Summer School on Software Engineering, Elba Island, September 2014.
- PI Alur gave a five-lecture course titled *Syntax Guided Synthesis* at Marktoberdorf Summer School in Germany in August 2014.
- PI Seshia taught a MOOC on Cyber-Physical Systems on edX in May-June 2014. This is the first MOOC to use synthesis technology in a virtual lab auto-grader, and the first MOOC to bring formal methods to a broader audience.
- PI Seshia taught the course *Formal Methods for Engineering Education* at UC Berkeley in Spring 2014. Several course projects have generated interesting results, including contributions to AutomataTutor and CPSGrader. Course materials are publicly available [6].

CCC Visioning Workshop on Computer-Aided Personalized Education

ExCAPE research on personalized education theme prompted us to put together a proposal to Computing Community Consortium (CCC) for a visioning activity on this topic. The proposal has been approved and the workshop will be organized on November 12 and 13. Our work has shown the benefits of applying tools for logical reasoning and machine learning to problems in personalized education in specific courses. The goal of the planned workshop is to build on this momentum to develop a long-term research agenda in collaboration with researchers in human-computer interaction and experts in education and learning. The workshop organizers include ExCAPE PIs Alur, Bodik, Hartmann, Solar-Lezama, and Vardi, and researchers in education, Baraniuk (Rice), Kafai (Penn), Karpicke (Purdue), and Thielle (Stanford).

Conferences and Workshops Organization, Awards and other Notable Items

- Bodik – steering committee member of NSF XPS workshop exploiting parallelism and scalability 2015.
- Bodik – general co-Chair of SNAPL 2015, the inagural summit on advances in programming languages.

- Bodik – general chair of POPL 2016, ACM SIGPLAN-SIGACT symposium on principles of programming languages.
- Bodik – steering committee member of ASPLOS 2014, inter. conf. on architectural support for programming languages and operating systems.
- Kavraki – general chair of Robotics: Science and Systems (RSS), 2015.
- Kavraki – program chair of Robotics: Science and Systems (RSS), 2014.
- Kavraki – Honorary Member, Artificial Intelligence Society of Greece, 2014.
- Kress-Gazit – Participant of NAE US Frontiers of Engineering, 2014.
- Kress-Gazit – general chair of the formal methods in robotic workshop at RSS 2014.
- Lafortune received the 2013-14 Vulcans Education Excellence Award from the College of Engineering at the University of Michigan.
- Lafortune received the 2014-15 Herbert Kopf Service Excellence Award from the College of Engineering at the University of Michigan.
- Lafortune – co-organizer of the Dagstuhl seminar on *Non-Zero Sum Games and Control*, 2015. Co-PIs Lafortune, Solar-Lezama, and Tripakis attended and made presentations at the seminar.
- Parthasarathy – co-chair of the SYNT 2015 workshop on Synthesis, co-held with CAV 2015.
- Seshia – program co-chair of Conference on Verified Software: Theories, Tools, and Experiments (VSTTE), 2015.
- Vardi – General Chair, 2014 Federated Logic Conference, Vienna, Austria.

Undergraduates and High-school Outreach

- PI Zdancewic is teaching programming to 3rd-6th graders as part of the University City's Arts League after school program.
- Two undergraduate students (Caleb Voss and Konstantinos Varvarezos) were included in the group of Kavraki and were exposed to the research material in ExCAPE.
- PI Kavraki is the mentor for Monica Sosa (2014-2015), in Empowering Leadership Alliance, Rice (ELA-RICE), which is a program that promotes and sustains a diverse community of science and engineering students at Rice and supports them as they become academic leaders in their respective fields.
- PI Kavraki is a faculty advisor for CSters — undergraduate women in computer science.
- PI Vardi is the research mentor for two undergraduate students (Sean Doyle and Jeffrey Dudek) in the area of constrained sampling.
- PI Alur worked with Penn undergraduate student Adam Freilich leading to publication [18]. Adam will be joining PhD program in computer science at Columbia University in Fall 2015
- PI Alur worked with Penn undergraduate student Matt Weaver during Summer and Fall 2014. Matt reimplemented the user interface for AutomataTutor and extended its functionality to handle problems about NFAs. As described in section 4.5, this tool is now used at over ten universities around the world.
- PI Alur is organizing a session titled *Design of Cyber-Physical Systems: From Pacemakers to Driverless Cars and Beyond* in Penn Engineerings's WICS Highschool Day for Girls (scheduled for April 10, 2015, with 200 girls registered for the event).
- PI Alur is giving a talk in the *Verification Mentoring Workshop* colocated with CAV 2015. The goal of the workshop is to encourage students to pursue research careers in formal methods.

Invited Talks

- *Model Synthesis: New Challenges for Model-Based Design*; Alur ;
  Toyota Workshop on Industrial Cyber-Physical Systems, Los Angeles, Dec 2014.
- *Regular functions*; Alur;
  Workshop on software correctness and reliability, ETH Zurich, Oct 2014;
  Descriptional Complexity of Formal Systems, Waterloo, Canada, June 2015.
- *Irrational Rationale Capture: Scaling Up Programmer's Apprentice While Holding Your Nose*; Bodik;
  NDIST 2014: New Directions in Software Technology, Caneel Bay, St. John, Nov 2014.
- *Survivalist Programming Models and Compilation*; Bodik;
  ISAT Survivalist Computing and Communications Study, Warrenton, VA, Nov. 2014.
- *Faster, Higher-Level, Stronger: Programming with Solver-Aided Languages*; Bodik;
  Keynote, at ISSTA Doctoral Symposium San Jose, CA, July, 2014 .

- *Constructing programming models and compilers for minimalistic low-power architectures*; Bodik;
  Ultra-Low Power Computing Workshop 2014, Redmond, WA, July 2014.
- *Synthesizing Web Broswer Layout Engines*; Bodik;
  Seminar for Servo Browser Developers, Mozilla, San Fancisco, CA, July 2014.
- *Adventures in Program Synthesis from synthesis to solver-aided languages*; Bodik;
  MPI-SW, Saarbrucken, Germany, May 2014.
- *Modeling Biology with Solver-Aided Languages*; Bodik;
  EPFL, Lausanne, Switzerland, May 2014.
- *Overview of Program Synthesis Research at UC Berkeley*; Bodik;
  Chaperone Project Retreat, Santa Cruz, CA, May 2014.
- *Synthesizing Inference Solvers with Partial Evaluation*; Bodik;
  Onsite review for DARPA PPAML, Cambridge, MA, April 2014.
- *Update on Berkeley HTML5 Research*; Bodik;
  SSG Research Program Review, Santa Clara, CA, March, 2014.
- *HW Accelerated Data Visualization on Tizen*; Bodik;
  Research Review, Samsung, Santa Clara, CA, March 2014.
- *Computed Aided Development for Mobile Applications*; Bodik;
  Research Review, Samsung, Santa Clara, CA, March 2014.
- *Design Methodology*; Bodik;
  NSF Expedition PI Onsite visit, Berkeley, CA, March 2014.
- *Prototyping and Just-In-Time Learning Tools*; Hartmann;
  FX Pal, Palo Alto, Aug 2014.
- *Q&A In Online Education: Experimental results*; Hartmann;
  ACM EC Workshop, Palo Alto, FX Pal, Palo Alto, June 2014.
- *Scaling Active Learning Activities* Hartmann;
  Rice University, Dec 2014.
- *Programming Systems meet Human-Computer Interaction*; Hartmann;
  Chaperone Retreat, Santa Cruz, Dec 15 2014.
- *Planning for Physical Systems*; Kavraki;
  Distinguished Lecture, Department of Computer Science, Brown University, Dec. 2014.
- *Planning for Complex High-Level Missions*; Kavraki;
  IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, Sep. 2014.
- *Computation of Robot Motion Plans from Complex High-Level Specifications*; Kavraki;
  National Center for Scientific Research, Demokritos, Athens, Greece, July 2014.
- *Temporal Motion Planning for Complex Dynamical Systems: Progress and Challenges*; Kavraki;
  The 5th Workshop on Formal Methods for Robotics, Robotics: Science and Systems, Berkeley, CA, July 2014.
- *Planning for Complex Physical Systems*; Kavraki;
  Hellenic Conference on Artificial Intelligence, Ioannina, Greece, May 2014.
- *Computation of Robot Motion Plans from Complex High-Level Specifications*; Kavraki;
  Lockheed Martin Maryland Robotics Series, University of Maryland, College Park, MD, May 2014.
- *Reasoning for Complex Physical Systems*; Kavraki;
  IRIM Seminar, School of Interactive Computing, Georgia Institute of Technology, Atlanta, March 2015.
- *Reasoning for Complex Systems: from Robots to Biomolecules*; Kavraki;
  Colloquium Lecture, Computer Science, USC Viterbi School of Engineering, February 2015.
- *Control and Diagnosis of Discrete Event Systems: Some Recent Trends*; Lafortune;
  General Electric Corporate Research, June 2014;
  University of Utah, June 2014;
  Georgia Tech, September 2014;
  Boston University, November 2014.
- *Formal Methods for Lab-Based MOOCs: Cyber-Physical Systems and Beyond*; Seshia;
  Keynote, Workshop on Embedded Systems Education at ESWEEK 2014, New Delhi, India, Oct. 2014.
- *Lab-based MOOCs: Cyber-Physical Systems, Robotics, and Beyond*; Seshia;
  National Instruments (NI) Week, Austin, TX, Aug. 2014.
- *Human-in-the-Loop Robotics: Specification, Verification, and Synthesis*; Seshia;
  Workshop on Formal Methods for Robotics and Automation (FMRA/RSS), Berkeley, CA, July 2014

- *A Logical Revolution*; Vardi;
  Workshop on Logic in Computer Science and Artificial Intelligence; University of Rome La Saplenza, Dec. 2014.
- *Compositional Temporal Synthesis*; Vardi;
  Seminar on Synthesis from Components, Schloss Dagstuhl, June 2014;
  Workshop on Reactive Systems, Federated Logic Conference, Vienna, July 2014;
  Keynote, 6th NASA Formal Methods Symposium, Clear Lake, TX, April 2014;
  From Programs to Systems, The Systems Perspective in Computing, ETAPS Workshop in honor of Joseph Sifakis, Grenoble, April, 2014.
- *SAT Sampling and Counting: From Theory to Practice*; Vardi;
  Keynote, Haifa Verification Conference, Haifa, Israel, November 2014.

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.

- Lafortune visited GE Corporate Research in Niskayuna, NY (June 2014) and UTC Aerospace Systems in Rockford, IL (November 2014) for technical discussions on synthesis problems in the area of Cyber-Physical Systems.
- Toyota organized a workshop on Industrial cycber-physical systems in Los Angeles in Dec 2014. A number of ExCAPE PIs, Alur, Tabuada, Tripakis, and Seshia participated in the meeting and gave talks.
- 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.
- A number of Penn students funded by ExCAPE did summer internships: Salar Moarref (Toyota, Spring 2014) and Mukund Raghothaman (Microsoft Research, Summer 2014).

Summary of the Mentoring Activities Conducted for Postdoc Researchers

- PI Parthasarathy mentored Dr. Daniel Neider in the Fall of 2015. He worked on bringing him up to speed on the style of research in the US, the focus on solving practical problems, and the open challenging problems in synthesis. Daniel worked with Parthasarathy on two research projects — one on building invariant synthesis using machine learning and the second on lifting learning techniques from invariant synthesis to much more general synthesis.

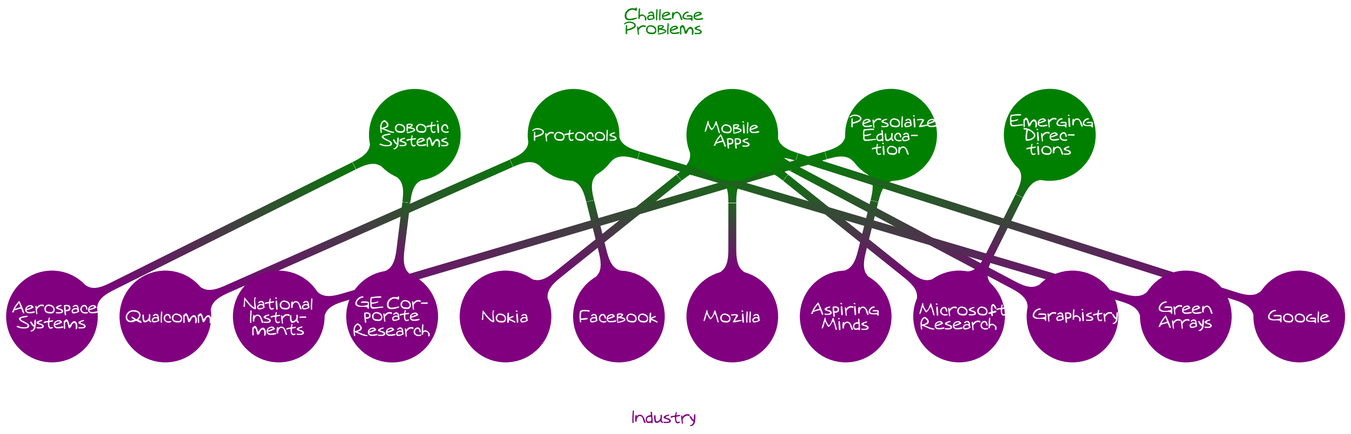

Fig. 1. Outreach to Industry

- PIs Foster and Solar-Lezama co-mentored Dr. Xiaokang Qiu. He continued the work on developing a new methodology for synthesizing models of complex platforms like Android, and applying his prior work on Natural Proofs to the problem of synthesizing provably correct data-structure manipulations. Xiaokang has also played a crucial role in mentoring undergraduate student Jeevana Priya Inala on the topic of synthesis of language desugaring functions. Additionally, Xiaokang has been helping PI Solar-Lezama to develop a new course on program synthesis at MIT which is largely based on ExCAPE research.
- PIs Seshia and Pappas co-mentored Dr. Indranil Saha on a novel project on compositional synthesis of multi-robot motion plans. They are also advising him on his search for a tenure-track faculty position.
- PIs Alur and Tripakis are co-mentors for Dr. Christos Stergiou who is working on synthesis of distributed protocols. Christos spend time both at Penn and at UC Berkeley over last two years.
- PI Vardi is advising Dr. Dror Fried and involves him in grant-proposal writing and collaboration with IIT Bombay. Dror is working on topics related to synthesis for robotic systems and SAT sampling.

Alumni

- Alvin Cheung, Faculty, University of Washington
- Eric Dallal, Postdoc, UCLA
- Alexander Gurney, Comcast
- Ruediger Ehlers, Faculty, University of Bremen in Germany
- Sela Mador-Haim, Coverity
- Arun Raghavan, Oracle Labs
- Matthias Rungger, Postdoc, Technical University Munich
- Rishabh Singh, Microsoft Research
- Emina Torlak, Faculty, University of Washington
- Anduo Wang, Postdoc, UIUC
- Yi-Chin Wu, Postdoc, UC Berkeley - University of Michigan
- Peter-Michael Osera is finishing his dissertation titled *Program synthesis with types* under the supervision of PI Zdancewic, and will join the faculty of Grinnell College starting Fall 2015.

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 two appointed lead PIs guiding research directions on that theme. The associate director, Fisman, is in charge of fostering collaborations, organizing ExCAPE webinars and meetings, putting together reports and presentations, and co-organizing the competition on Syntax Guided Synthesis.

A main mechanism for fostering collaborations under ExCAPE, is its post-doctoral researchers, which are mentored by at least two PIs from two different institutions. Xiaokang Qiu, Indranil Saha, Christos Stergiou and Daniel Neider have been working on collaborative projects under this scheme (see Summary of mentoring activities). For the upcoming year we hope to recruit two such jointly-mentored post-doctoral researchers, one with an expertise in human-computer interaction.

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: John Field (Google), Limor Fix (Intel), Patrice Godefroid (Microsoft), Aarti Gupta (NEC), Datta Godbole (Honeywell), Andreas Kuehlmann (Coverity), Pieter Mosterman (Mathworks), Mark Wegman (IBM) and Pamela Zave (AT&T). We are fortunate to have Gupta, Godefroid, Kuehlmann, Mosterman, and Wegman following ExCAPE closely, participating in many ExCAPE meeting, and continuously providing us with valuable feedback. Satish Chandra, Samsung Electronics, is joining the industrial advisory board and will participate in June 2015 PI meeting. Furthermore, as indicated in the previous section, we have also established collaborations with other researchers at Aerospace Systems, Google, National Instruments, Qualcomm, Facebook, Microsoft Research, Nokia, AspiringMinds, GE Corporate Research, Graphistry, GreenArrays and Mozilla.

In the review meeting we received valuable comments and suggestions from the review team: Frank Pfenning (CMU), Viktor Kuncak (EPFL), Bill Griswold (UCSD), John Reppy (U. Chicago), Ewen Denney (NASA) and John Harrison (Intel). Pursuing these, we have proposed a visioning workshop on computer-aided personalized education to the Computing Community Consortium (CCC) hoping to galvanize a wider community around it and to articulate a broad and long-term research agenda. We also plan to invite SE researchers to ExCAPE meeting in June 2015, and organize a tutorial at either ICSE or FSE on program synthesis, as envisioned by ExCAPE, to facilitate a dialog with the SE community.

8 Collaborations

We have used frequent teleconferences as a means of fostering collaboration. Teleconferences are organized by groups of PIs interested in a common thread. For example, the groups working on multicore systems and distribted protocosl from Berkeley and Penn have weekly teleconferences; all robotics groups joined this year for a series of teleconferences; the robotics groups at Rice/UCLA have weekly teleconferences; the robotics groups at Cornell/Rice have bi-monthly teleconferences; and the groups involved in SyGuS at Berkeley/MIT/Penn/UIUC have bi-weekly teleconferences. The events that bring all the PIs together include the Annual PI meetings and the monthly ExCAPE webinars as discussed earlier. Collaborations across themes, disciplines and institutions are depicted in Figures 2, 3 and 4. The following collaborations where made possible due to ExCAPE:

- Alur and Tripakis are collaborating on synthesis of distributed protocols. For this project, there is a weekly telecon. This has resulted in two joint publications over the past year.
- Alur is on PhD dissertation committee of Penn PhD student Peter-Michael Osera, advised by Zdancewic.
- Alur and Vardi collaborated to write a survey on success of formal methods [19].
- Ayca Balkan, Ph.D. student in Tabuada's group at UCLA visited Vardi's group at Rice University to discuss a joint project on the analysis and reduction of GR(1) into games.
- Postdocs Ankur Mehta and Javier Alonso-Mora from Daniela Rus's group (MIT) within the expedition on Printable Programmable Machine visited Kress-Gazit's group. Two papers on topics related to ExCAPE where submitted as a result.
- Bi-weekly Skype meetings between Morteza Lahijanian, postdoc at Rice and Matthias Ruggerm, postdoc at UCLA on a the joint project titled, "Sampling-based Construction of Symbolic Models for Control Systems" continued despite Dr. Rugger's move to UT Munich.
- Bodik was an external examiner for Eva Darulova's thesis defense (EPFL), September 2014
- Bodik was on the qualifying exam external committee member, Mirco Giacobbe, IST.
- Bodik has been collaborating with Kirsten Beck (UW), Ernest Fraenkel (MIT), Anthony Gitter (Wisconsin), Jasmin Fisher (Cambridge, MSR), Aaron McKenna (UW), Nir Piterman (Leicester), Saurabh Srivastava (20n) and Alejandro Wolf-Yadlin (UW) on topics related to Synthesis in Biology.

- The collaboration of PIs Kavraki, Kress-Gazit and Vardi on the topic of synthesis for hybrid systems with maximal satisfaction guarantees resulted in a manuscript submitted to IEEE Transactions on Robotics.
- Kress-Gazit and Tabuada's group collaborated on the topic of robustness in robot motion planning, and a paper is under review at IJRR.
- Kuldeep Meel, from Vardi's group visited Seshia's group in June 2014, to collaborate about the topic of SAT sampling.
- Lafortune, Tripakis, and Vardi and Ehlers continue collaborating on the work on bridging the gap between reactive synthesis and supervisory control.
- Lafortune's PhD student Yi-Chin Wu graduated in August 2014 and joined the TerraSwarm Research Center as a post-doctoral researcher. This is leading to synergy between ExCAPE and TerraSwarm on synthesis problems that arise in the enforcement of security and privacy in networked computing systems. In this new effort, Lafortune and Wu are collaborating with Seshia of ExCAPE and Edward Lee, Project Director of TerraSwarm.
- Lafortune's PhD student Eric Dallal graduated in August 2014 and joined the group of PI Tabuada at UCLA as a post-doctoral researcher.
- Lafortune hosted Anne-Kathryn Schmuck of TU-Berlin for technical discussions on the ExCAPE project, November 24-25, 2014.
- Lafortune is co-organizing with Solar-Lezama, Zdancewic, and Fisman the 2015 ExCAPE Summer School to be held in June at MIT.
- Loo and Alur are co-advising PhD student Yifei Yuan.
- Parthasarathy is collaborating with Christof Loding (RWTH, Aachen) on topics related to invariant synthesis.
- Seshia is on the PhD dissertation committee of Penn PhD student Yifei Yuan, co-advised by Alur and Loo.
- Tripakis is on the PhD dissertation committee of Penn PhD student Abhishek Udupa, co-advised by Alur and Martin.
- Vardi and Kavraki co-advise Morteza Lahijanian (postdoc) and Keliang He (PhD Student) at Rice University.
- Vardi is on PhD dissertation committee of Penn PhD student Mukund Raghothaman, advised by Alur.
- Yasser Shoukry, from Tabuada's group visited Seshia's group June-Aug 2014 to collaborate on the topic of robust synthesis.
- Zdancewic is collaborating with David Walker (Princeton) on type- and example-driven program synthesis.

ExCAPE-ARiSE Collaboration

ExCAPE and the Austrian project RiSE on Rigorous System Engineering are collaborating in various manners.

- The annual ExCAPE meeting that took place March 2014 in Berkeley was joint with RiSE. The RiSE researchers Roderick Bloem (TU Graz), Krishnendu Chatterjee (IST), Radu Grosu (TU Vienna), Swen Jacobs (TUGraz), Christoph Kirsch (Salzburg), Arjun Radhakrishna (IST Austria), Georg Weissenbacher (TU Wien) and Florian Zuleger (TU Vienna) gave talks in this meeting.
- Dr. Arjun Radhakrishna, who did his PhD at IST Austria, under the supervision of Thomas A. Henzinger, is now an ExCAPE postdoc, mentored by Rajeev Alur.
- RiSE members will attend the ExCAPE summer school at MIT, in which Roderick Bloem (Graz) will give a tutorial on reactive synthesis.
- ExCAPE members will attend the winter school to be organized by RiSE early 2016 in Europe.

References

1. Automatatutor website. http://www.automatatutor.com.
2. Chloropyll sorce code. https://github.com/mangpo/chlorophyll.
3. Complan video. https://www.youtube.com/watch?v=4IHR_9SgK40.
4. CPSgrader website. http://cpsgrader.org/.
5. Desuma websit se. https://wiki.eecs.umich.edu/desuma/index.php/DESUMA.
6. Formal methods for engineering education course material. http://www.eecs.berkeley.edu/ sseshia/fmee/.
7. Problem generator website. http://50.112.174.116/.
8. Ringer source code. https://github.com/sbarman/webscript.
9. Rosette source code. https://github.com/emina/rosette.

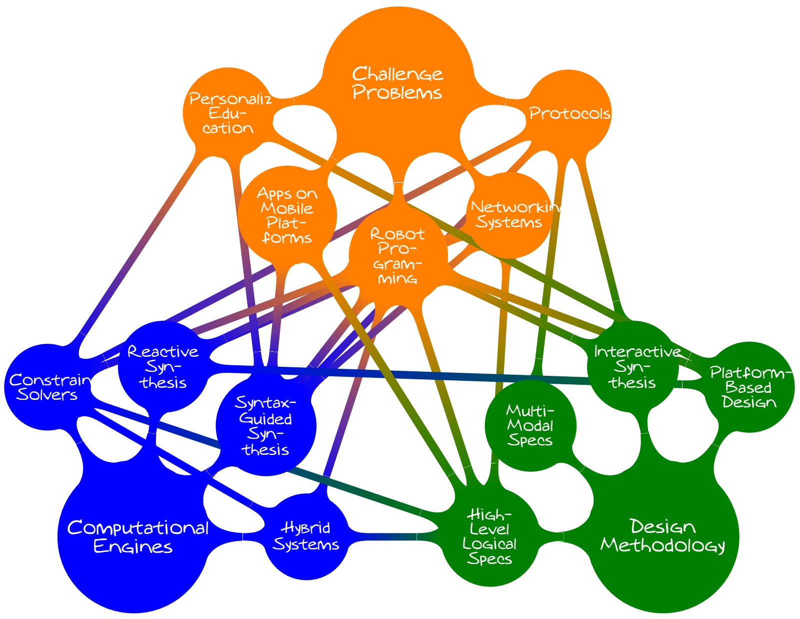

Fig. 2. Collaboration across themes

10. Rosette website. http://homes.cs.washington.edu/ emina/rosette/.
11. Umdes website. https://wiki.eecs.umich.edu/desuma/index.php/UMDES_Software_Library.
12. WebCombine source code. https://github.com/schasins/structured-data-scraping-extension.
13. WebCombine video. https://www.youtube.com/watch?v=1AkS89VWJgE.
14. R. Alur, R. Bodik, Dallal, D. Fisman, P. Garg, H. Kress-Gazit, G. Juniwal, M. Martin, P. Madhusudan, M. Raghothaman, S. Saha, S. A. Seshia, R. Singh, Armando Solar-Lezama, E. Torlak, and A. Udupa. Syntax-guided synthesis. In *Proceedings of the NATO on Dependable Software Systems Engineering, Marktoberdorf, Germany.*, 2015. under review.
15. R. Alur, R. Bodik, 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*, pages 1–17, 2013.
16. R. Alur, P. Cerny, and A. Radhakrishnan. Synthesis through unification. Submitted. 2015.
17. R. Alur, L. D'Antoni, and M. Raghothaman. Drex: A declarative language for efficiently evaluating regular string transformations. In *42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL*, pages 125–137, 2015.
18. R. Alur, A. Freilich, and M. Raghothaman. Regular combinators for string transformations. In *23rd EACSL Annual Conference on Computer Science Logic (CSL) and the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14*, 2014.
19. R. Alur, T.A. Henzinger, and M.Y. Vardi. Theory in practice for system design and verification. In *SIGLOG News*, 2015.
20. R. Alur, M. Martin, M. Raghothaman, C. Stergiou, S. Tripakis, and A. Udupa. Synthesizing finite-state protocols from scenarios and requirements. In *10th Haifa Verification Conference (HVC)*, 2014.
21. R. Alur, S. Moarref, and U. Topcu. Counter-strategy-guided compositional synthesis for multi-agent systems. Submitted. 2015.
22. R. Alur, S. Moarref, and U. Topcu. Pattern-based refinement of interface specifications in reactive synthesis. In *21st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2015.
23. R. Alur, M. Raghothaman, C. Stergiou, S. Tripakis, and A. Udupa. Automatic completion of distributed protocols with symmetry. Submitted. 2015.
24. R. Alur and N. Singhania. Precise piecewise affine models from input-output data. In *14th ACM Intl. Conf. on Embedded Software (EMSOFT)*, 2014.
25. D. Angluin and D. Fisman. Learning regular omega languages. In *25th International Conference on Algorithmic Learning Theory (ALT)*, 2014.
26. A. Butez, S. Lafortune, and Y. Wang. State-partition-based control of discrete event systems for enforcement of regular language specifications. In *Proceeding of the 2014 IFAC World Congress*, 2014.
27. S. Chakraborty, D. J. Fremont, K. S. Meel, S. A. Seshia, and M. Y. Vardi. On parallel scalable uniform SAT witness generation. In *Proc. of TACAS, To appear*, 2015.
28. S. Chakraborty, D. J. Fremont, K .S. Meel, S. A. Seshia, and M.Y. Vardi. Distribution-aware sampling and weighted model counting for SAT. In *The Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

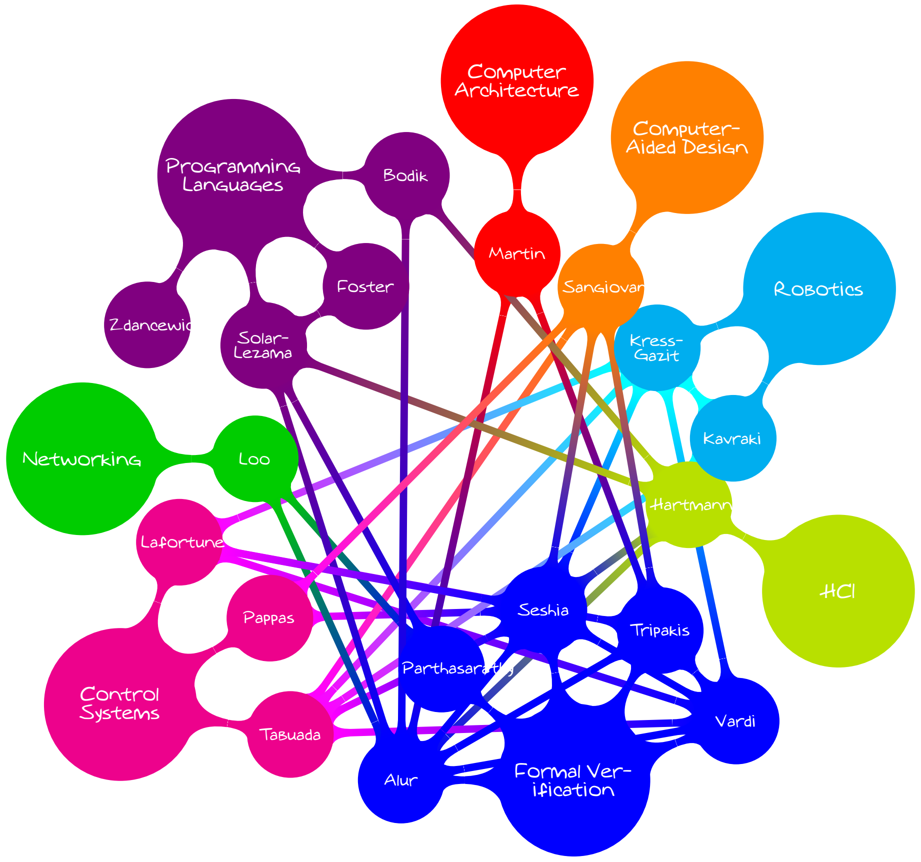

Fig. 3. Collaboration across disciplines

29. S. Chakraborty, K. S. Meel, and M. Y. Vardi. A scalable and nearly uniform generator of SAT witnesses. In *Proc. of CAV*, 2013.
30. S. Chakraborty, K. S. Meel, and M. Y. Vardi. A scalable approximate model counter. In *Proc. of CP*, pages 200–216, 2013.
31. S. Chakraborty, K. S. Meel, and M. Y. Vardi. Balancing scalability and uniformity in SAT witness generator. In *Proc. of DAC*, 2014.
32. K. Chatterjee, L. Doyen, S. Nain, and M. Y. Vardi. The complexity of partial-observation stochastic parity games with finite-memory strategies. In *Foundations of Software Science and Computation Structures*, pages 242–257. Springer, 2014.
33. A. Cheung, A. Solar-Lezama, and S. Madden. Optimizing database-backed applications with query synthesis. In *In Hans-Juergen Boehm and Cormac Flanagan, editors, PLDI*, pages 3âĂŞ–14, 2013.
34. E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune. Supervisory control for collision avoidance in vehicular networks using discrete event abstractions. Submitted for journal publication.
35. E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune. Supervisory control of systems under imperfect measurements with applications to vehicle control. Submitted for journal publication.
36. L. D'Antoni and R. Alur. Symbolic visibly pushdown automata. In *26th International Conference on Computer-Aided Verification (CAV)*, 2014.
37. L. D'Antoni, Dileep K., R. Alur, S. Gulwani, M. Viswanathan, and B. Hartmann. How can automatic feedback help students construct automata? *TOCHI: ACM Transactions on Computer-Human Interaction*, 2015.
38. J. A. DeCastro, R. Ehlers, M. Rungger, A. Balkan, P. Tabuada, and H. Kress-Gazit. Dynamics-based reactive synthesis and automated revisions for high-level robot control. *International Journal of Robotics Research, under review*.
39. J. A. DeCastro and H. Kress-Gazit. Synthesis of nonlinear continuous controllers for verifiably-correct high-level. *Journal of Reactive Behaviors*, 2014.
40. 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*, 2014.
41. J. Fisher, N. Piterman, and R. Bodik. Towards synthesizing executable models in biology. In *Frontiers in Bioengineering and Biotechnology*, 2014.
42. J. Galenson, R. Bodik, and K. Sen. CodeHint: Dynamic and interactive synthesis for modern IDEs (refereed presentation). In *Future Programming (FP)*, 2014.

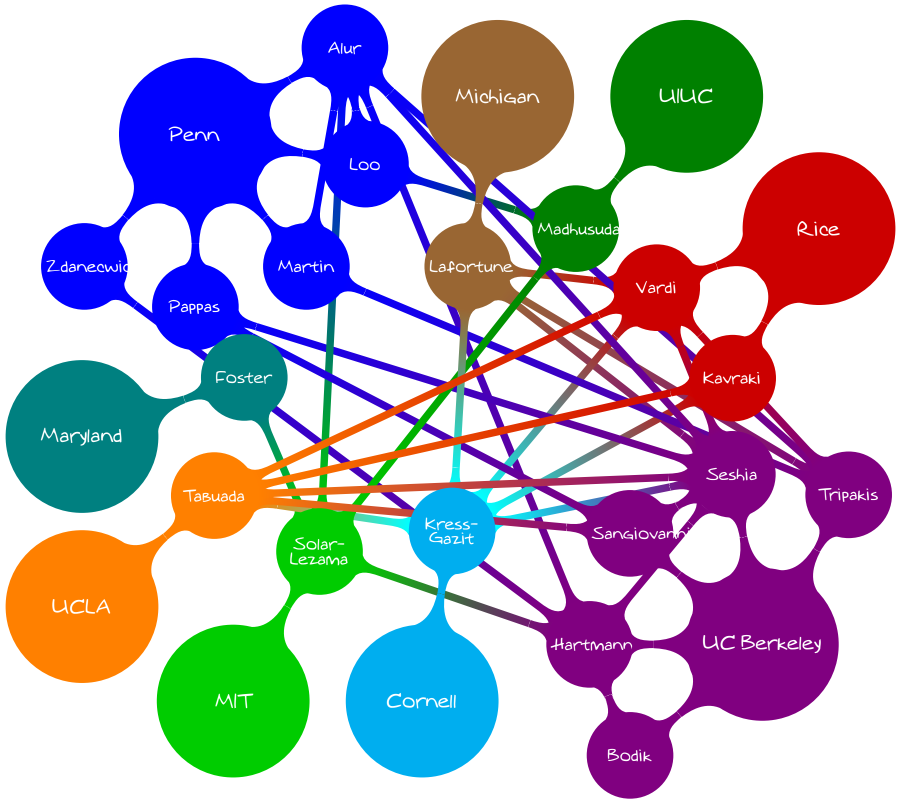

Fig. 4. Collaboration across institutions

43. J. Galenson, P. Reames, R. Bodik, B. Hartmann, and K. Sen. CodeHint: Dynamic and interactive synthesis of code snippets. In *36th International Conference on Software Engineering (ICSE)*, 2014.
44. T. Hottelier, R. Bodik, and K. Ryokai. Programming by manipulation for layout. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2014.
45. J. Inala, X. Qiu, B. Lerner, and A. Solar-Lezama. Type assisted synthesis of programs with algebraic data types. submitted.
46. J. Jeon, X. Qiu, J. S. Foster, and A. Solar-Lezama. Pasket: Synthesizing framework models with design patterns. under review.
47. J. Jeon, X. Qiu, A. Solar-Lezama, and J. S. Foster. Adaptive concretization for parallel program synthesis, 2015. under review.
48. G. Juniwal. CPSGrader: Auto-grading and feedback generation for cyber-physical systems education. Master's thesis, EECS Department, University of California, Berkeley, Dec 2014.
49. G. Juniwal, A. Donzé, J. C. Jensen, and S. A. Seshia. CPSGrader: Synthesizing temporal logic testers for auto-grading an embedded systems laboratory. In *Proceedings of the 14th International Conference on Embedded Software (EMSOFT)*, 2014.
50. G. Juniwal, S. Jain, A. Donzé, and . A. Seshia. Clustering-based active learning for CPSGrader. In *L@S 2015: Second (2015) ACM Conference on Learning @ Scale Proceedings*, 2015.
51. A. S. Köksal, A. Gitter, K. Beck, A. McKenna, S. Srivastava, N. Piterman, R. Bodik, A. Wolf-Yadlin, E. Fraenkel, and J. Fisher. Synthesizing signaling pathways from temporal phosphoproteomic data (refereed presentation). In *RECOMB/ISCB Conference on Regulatory and Systems Genomics*, 2014.
52. M. Lahijanian, S. Almagor, D. Fried, L. E. Kavraki, and M. Y. Vardi. This time the robot settles for a cost: A quantitative approach to temporal logic planning with partial satisfaction. In *The Twenty-Ninth AAAI Conference*, Austin, TX, Jan. 2015.
53. M. Lahijanian, M. Maly, D. Fried, L. Kavraki, H. Kress-Gazit, and M. Vardi. Iterative temporal planning in uncertain environments with partial satisfaction guarantees. *IEEE Transactions on Robotics, under review*.
54. R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki. Asymptotically optimal stochastic motion planning with temporal goals. In *The Eleventh International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, Istanbul, Turkey, Aug. 2014.
55. R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki. Fast stochastic motion planning with optimality guarantees using local policy reconfiguration. In *IEEE Conference on Robotics and Automation*, pages 3013–3019, Hong Kong, China, May 2014.
56. R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki. Optimal and efficient stochastic motion planning in partially-known environments. In *The Twenty-Eighth AAAI Conference on Artificial Intelligence*, Quebec City, Canada, Jul. 2014.
57. S. Maniatopoulos, M. Blair, C. Finucane, and H. Kress-Gazit. Open-world mission specification for reactive robots. In *IEEE International Conference on Robotics and Automation*, 2014.

58. L. A. Meyerovich, M. E. Torok, E. Atkinson, and R. Bodik. Superconductor: A language for big data visualization. In *Workshop on Leveraging Abstractions and Semantics in High-performance Computing (LASH-C)*.
59. F. Mogavero, A. Murano, and M. Y. Vardi. Relentful strategic reasoning in alternating-time temporal logic. *Journal of Logic and Computation*, 2014.
60. P. Nuzzo, A. Iannopollo, S. Tripakis, and A. L. Sangiovanni-Vincentelli. Are interface theories equivalent to contract theories? In *12th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2014.
61. P-M. Osera and S. Zdancewic. Type-and-example-driven program synthesis. In *Programming Language Design and Implementation (PLDI)*, 2015. To appear.
62. P. M. Phothilimthana, T. Jelves, R. Shah, N. Totla, S. Chasins, and R. Bodik. Chlorophyll: Synthesis-aided compiler for low-power spatial architectures. In *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, 2014.
63. V. Preoteasa and S. Tripakis. Refinement calculus of reactive systems. In *Proceedings of the 14th ACM & IEEE International Conference on Embedded Software (EMSOFT'14)*, 2014.
64. V. Raman and H. Kress-Gazit. Synthesis for multi-robot controllers with interleaved motion. *IEEE International Conference on Robotics and Automation*, 2014.
65. V. Raman, N. Piterman, C. Finucane, and H. Kress-Gazit. Timing semantics for abstraction and execution of synthesized high-level robot control. *IEEE Transactions on Robotics, accepted*.
66. Y. Shoukry, A. Puggelli, P. Nuzzo, A. Sangiovanni-Vincentelli, S. Seshia, and P. Tabuada. Sound and complete state estimation for linear dynamical systems under sensor attack using satisfiability modulo theory solving. In *Proceedings of the 2015 American Control Conference*, 2015. To appear.
67. Solar-Lezama and Xiaokang Qiu. Mit course-2015-6.885, introduction to principles and practice of software synthesis. https://stellar.mit.edu/S/course/6/sp15/6.885/index.html.
68. J. Stanley, H. Liao, and S. Lafortune. SAT-based control of concurrent software for deadlock avoidance. Submitted for journal publication.
69. E. Torlak and R. Bodik. A lightweight symbolic virtual machine for solver-aided host languages. In *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, 2014.
70. E. Torlak and R. Bodik. A lightweight symbolic virtual machine for solver-aided host languages. In *CM SIGPLAN Conference on Programming Language Design and Implementation, PLDI*, 2014.
71. K.W. Wong, R. Ehlers, and H. Kress-Gazit. Correct high-level robot behavior in environments with unexpected events. In *Proceedings of Robotics: Science and Systems*, 2014.
72. Y. C. Wu and S. Lafortune. Synthesis of insertion functions for enforcement of opacity security properties. In *Automatica, Vol. 50 No. 5*, pages 1336–1348, 2014.
73. Y. C. Wu, K.A. Sankararaman, and S. Lafortune. Ensuring privacy in location-based services: An approach based on opacity enforcement. In *Proceedings of the 12th International Workshop on Discrete Event Systems*, 2014.
74. X. Yin and S. Lafortune. Codiagnosability and coobservability under dynamic observations: Transformation and verification. Submitted for journal publication.
75. X. Yin and S. Lafortune. Synthesis of maximally permissive supervisors for partially-observed discrete-event systems. Submitted for journal publication.
76. X. Yin and S. Lafortune. A general approach for synthesis of supervisors for partially-observed discrete-event systems. In *Proceeding of the 2014 IFAC World Congress*, 2014.
77. X. Yin and S. Lafortune. A new approach for synthesizing opacity-enforcing supervisors for partially-observed discrete-event systems. In *Proceedings of the 2015 American Control Conference*, 2015.
78. X. Yin and S. Lafortune. On the relationship between codiagnosability and coobservability under dynamic observations. In *Proceedings of the 2015 American Control Conference*, 2015.
79. Y. Yuan, R. Alur, and B. T. Loo. NetEgg: Programming network policies by examples. In *13th ACM Workshop on Hot Topics in Networks (HotNets-XIV)*, 2014.
80. Y. Yuan, D. Lin, R. Alur, and B. T. Loo. Scenario-based programming of SDN policies. Submitted.