On the Teaching Dimension of Octagons for Formal Synthesis

Susmit Jha
UTRC, Berkeley
jhask@utrc.utc.com

Sanjit A. Seshia
EECS, UC Berkeley
sseshia@eecs.berkeley.edu

Xiaojin(Jerry) Zhu
CS Dept., University of Wisconin, Madison
jerryzhu@cs.wisc.edu

Formal synthesis is the process of generating a program satisfying a high-level formal specification. In recent times, effective formal synthesis techniques have been proposed based on the integration of inductive learning and formal methods. We refer to this class of methods that learn programs from examples as formal inductive synthesis. One of the key challenges in formal inductive synthesis is the identification of a sufficient set of examples that *induce* the correct program. In previous work, we have shown that the size of this sufficient set of examples is closely related to a notion from computational learning theory termed as the *teaching dimension*. However, not much is known about the teaching dimension of concepts that are relevant to formal verification and synthesis. In this paper, we present a result on the teaching dimension of *generalized octagons*, which are formed as conjunctions of unit two-variable-per-inequality constraints. Octagons have found widespread use in verification and synthesis problems. They are routinely used for abstract interpretation in programs as well as to express switching conditions for hybrid systems. We give a theorem about the minimum number of examples needed to teach octagons which in turn characterizes the complexity of formal inductive synthesis of octagons in a formal verification or synthesis setting.

1 Problem Definition

Formal inductive synthesis (FIS) [7] is the problem of generating a program satisfying a formal specification from a set of examples/observations. The observations are typically generated through the use of an "oracle" that can answer queries from a "learner." The learner aims to synthesize the program from examples without direct access to the entire specification. FIS has been shown to be effective in various settings, including program synthesis [12, 5], invariant generation [3, 10], requirement mining [8], and switching logic synthesis [6]. A particularly popular paradigm for solving FIS problems is *counterexample-guided inductive synthesis* (CEGIS) [12, 11] in which the oracle is implemented as a verification tool that can provide counterexamples if the current program generated by the learner does not satisfy the specification. Other approaches are also possible; see [7] for more details.

One measure of complexity of an FIS problem is the length of the the dialogue — the number of (query, response) exchanges — between oracle and learner. In our previous work [7], we showed that for any FIS problem involving an oracle that provides counterexamples, the length of this dialogue is bounded below by a quantity studied by the machine learning theory community called the *teaching dimension* (TD) [4]. The TD is a property of the class of programs being learned – also called the *concept class* in machine learning. We recall the formal definition of teaching dimension below:

Definition 1.1 *For a given concept class C and a target concept $c \in C$, we say that T is a teaching sequence for c if T is a sequence of positive and negative examples that uniquely identifies $c \in C$, that is, c is the only concept consistent with all the positive and negative examples in T. Let $T(c)$ denote all the teaching sequences for a target concept c. The teaching dimension $TD(C)$ is given as follows:* $TD(C) = \max_{c \in C} \left(\min_{\tau \in T(c)} |\tau| \right)$.

Submitted to:
SYNT 2016

As identified in our previous work [7], it is still a major outstanding challenge to compute the teaching dimension for various concept classes that are relevant for formal verification and synthesis. In this abstract, we would like to share some initial progress in this regard. Specifically, we have obtained a result on the TD of *generalized octagons*. These are logical formulae obtained as the conjunction of unit two variable per inequality constraints. Octagons have found use in a variety of settings in verification, from being a very effective domain for abstract interpretation [1, 9, 2] to being useful for specifying switching conditions for hybrid systems [6]. We believe this result is new from the learning theory perspective and relevant for understanding the complexity of formal inductive synthesis of octagons in various formal verification and synthesis settings. In the next section, we summarize the key ideas.

2 Teaching Octagons

Consider a d dimensional hyperspace. Any point p in the space is a d-tuple with its i-th component denoted by $p(i)$. We consider linear inequalities or hyperplanes formed by at most two variable per inequality constraints, where the kth constraint is given as: $S_k(p) \leq c_k$ where $S_k(p) = \pm p(i) \pm p(j)$ or $S_k(p) = \pm p(l)$. The number of such distinct hyperplanes is bounded by $K = 2d(d-1) + 2d = 2d^2$. Next, we define *generalized octagons*.

Definition 2.1 *Generalized octagons in d dimensional hyperspace are convex closed polytopes formed by intersection of half-spaces corresponding to the above hyperplanes, that is,* $\texttt{Octagon}(c_1, c_2, \ldots, c_K) = \{p \mid \bigwedge_{k=1}^{K} S_k(p) \leq c_k\}$ *where* c_k *are finite constants .*

Definition 2.2 *We say that a hyperplane* $S_{k'}(p) \leq c_{k'}$ *in a generalized octagon is an inactive hyperplane if and only if* $\left(\bigwedge_{k=1,k\neq k'}^{K} S_k(p) \leq c_k\right) \Rightarrow S_{k'}(p) < c_{k'}$. *Given an octagon P formed by active hyperplanes* `Act`*, and having vertices V, the smallest subset of vertices* $V_{moc} \subseteq V$ *such that each hyperplane in* `Act` *is incident on atleast one vertex in* V_{moc}*, is called the minimal octagon-covering set (MOC) of vertices.*

We now state the main result on teaching octagons below that summarizes the size of the teaching set, that is, the number of examples needed to teach octagon concepts.

Theorem 2.1 *Given an octagon P, there exists a teaching set of size* $|V_{moc}| + |F|$*, where* $|V_{moc}|$ *is the number of vertices in the minimal octagon-covering set (MOC) and* $|F|$ *is the number of faces of the octagon. Moreover, this is the size of the minimum teaching set.*

We omit the proof for lack of space. We only remark that our result generalizes one by Goldman and Kearns [4] in their original paper about teaching dimension, where they reported that for *hyperboxes* (conjunctions of interval constraints, a special case of octagons), the TD is $2d+2$: note that for a hyperbox, V_{moc} is 2 and $|F|$ is $2d$.

3 Future Work

Our initial result is restricted to generalized octagons - geometric concepts formed by conjunction of unit two variable per inequality constraints. As the next step, it would be interesting to generalize the results to general polytopes, or to programs that involve linear operations. We also plan to consider a variant of teaching dimension which is aware of the learning algorithm used for generalizing from examples [13].

References

[1] Patrick Cousot & Radhia Cousot (1977): *Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*. In: *Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, ACM, pp. 238–252.

[2] Patrick Cousot, Radhia Cousot, Jerôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux & Xavier Rival (2005): *The ASTRÉE analyzer*. In: *Programming Languages and Systems*, Springer, pp. 21–30.

[3] Pranav Garg, Christof Löding, P Madhusudan & Daniel Neider (2014): *ICE: A robust framework for learning invariants*. In: *Computer Aided Verification (CAV)*, pp. 69–87.

[4] Sally A. Goldman & Michael J. Kearns (1992): *On the Complexity of Teaching*. *Journal of Computer and System Sciences* 50, pp. 303–314.

[5] Susmit Jha, Sumit Gulwani, Sanjit A. Seshia & Ashish Tiwari (2010): *Oracle-guided Component-based Program Synthesis*. ICSE '10, ACM, New York, NY, USA, pp. 215–224, doi:10.1145/1806799.1806833.

[6] Susmit Jha, Sumit Gulwani, Sanjit A. Seshia & Ashish Tiwari (2010): *Synthesizing Switching Logic for Safety and Dwell-Time Requirements*. In: *ICCPS*, pp. 22–31.

[7] Susmit Jha & Sanjit A Seshia (2015): *A theory of formal synthesis via inductive learning*. *arXiv preprint arXiv:1505.03953*.

[8] Xiaoqing Jin, Alexandre Donzé, Jyotirmoy Deshmukh & Sanjit A. Seshia (2013): *Mining Requirements from Closed-Loop Control Models*. In: *Proceedings of the International Conference on Hybrid Systems: Computation and Control (HSCC)*.

[9] Antoine Miné (2006): *The octagon abstract domain*. *Higher-order and symbolic computation* 19(1), pp. 31–100.

[10] Rahul Sharma & Alex Aiken (2014): *From Invariant Checking to Invariant Inference Using Randomized Search*. In: *26th International Conference on Computer Aided Verification (CAV)*, pp. 88–105.

[11] Armando Solar Lezama (2008): *Program Synthesis By Sketching*. Ph.D. thesis, EECS Department, University of California, Berkeley.

[12] Armando Solar-Lezama, Liviu Tancau, Rastislav Bodik, Sanjit Seshia & Vijay Saraswat (2006): *Combinatorial Sketching for Finite Programs*. *SIGOPS Oper. Syst. Rev.* 40(5), pp. 404–415.

[13] Xiaojin Zhu (2015): *Machine Teaching: An Inverse Problem to Machine Learning and an Approach Toward Optimal Education*. *AAAI Conference on Artificial Intelligence*.