Dimensions in Program Synthesis

(Tutorial)

Sumit Gulwani

Microsoft Research

Redmond, WA, 98052

Email: sumitg@microsoft.com

Abstract

Program Synthesis, which is the task of discovering programs that realize user intent, can be useful in several scenarios: discovery of new algorithms, helping regular programmers automatically discover tricky/mundane programming details, enabling people with no programming background to develop scripts for performing repetitive tasks (end-user programming), and even problem solving in the context of automating teaching.

In this tutorial, I will describe the three key dimensions that should be taken into account in designing any program synthesis system: expression of user intent, space of programs over which to search, and the search technique [1]. (i) The user intent can be expressed in the form of logical relations between inputs and outputs, input-output examples, demonstrations, natural language, and inefficient or related programs. (ii) The search space can be over imperative or functional programs (with possible restrictions on the control structure or the operator set), or over restricted models of computations such as regular/context-free grammars/transducers, or succinct logical representations. (iii) The search technique can be based on exhaustive search, version space algebras, machine learning techniques (such as belief propagation or genetic programming), or logical reasoning techniques based on SAT/SMT solvers.

I will illustrate these concepts by brief description of various program synthesis projects that target synthesis of a wide variety of programs such as standard undergraduate textbook algorithms (e.g., sorting, dynamic programming), program inverses (e.g., decoders, deserializers), bitvector manipulation routines, deobfuscated programs, graph algorithms, text-manipulating routines, geometry algorithms etc.

REFERENCES

[1] S. Gulwani. Dimensions in program synthesis. In *ACM Symposium on PPDP*, 2010.