Automatic Generation of Program Invariants from Traces

Software bugs are a persistent feature of daily life—crashing web browsers, allowing cyber attacks, and distorting the results of scientific computations. One approach to improving software quality uses program invariants—mathematical descriptions of program behaviors—to verify programs, detect bugs, and repair errors. Although three decades of research have developed techniques for invariant generation, current methods lack support for several important classes of invariants, such as properties about arrays. As a result, we lack the ability to conduct precise analysis of most programs involving this popular data structure. The talk will describe DIG, a tool that dynamically infers invariants from program execution traces and statically verifies candidate invariants against program source code. DIG combines mathematics and formal methods to discover several challenging but useful classes of program invariants, including nonlinear polynomial relations, which are fundamental to many scientific applications, disjunctive invariants, which express branching behaviors in programs, and properties about multi-dimensional arrays, which appear in many practical applications. The talk will describe theoretical and empirical results showing that DIG can efficiently and accurately find many important invariants in real-world uses, e.g., polynomial properties in numerical algorithms and array relations in a full AES encryption implementation.

ThanhVu Nguyen received BS and MS degrees in computer science from Penn State University and is currently a Ph.D. candidate in Computer Science at the University of New Mexico. His research interests lie in the intersection of software engineering and programming languages, with a particular focus on using static and dynamic analyses for automatic invariant generation and program repair.